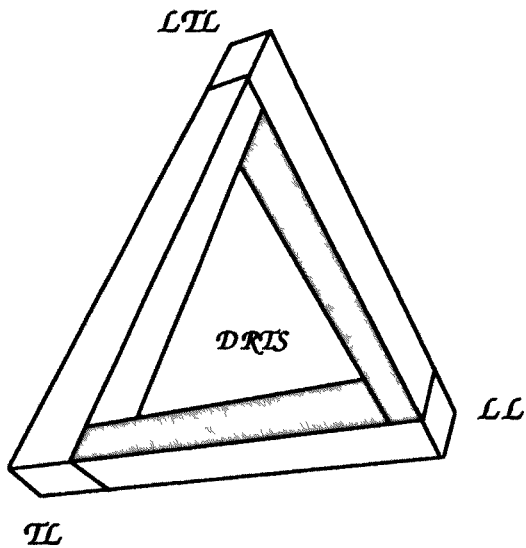

LTL
DRTS
LL
TL

Locative Temporal Logic
and
Distributed Real-Time Systems

—Specification—

Locatieve Temporele Logica
en
Gedistribueerde Real-Time Systemen

—Specificatie—

ISBN 90 373 0257 2 / CIP

COVER GRAPHIC It illustrates in an abstract way both a language space with the three logical languages investigated in this thesis and a distributed real-time system constructed from three processors. The idea of a language space is due to W.P. Koole.

The graphic itself depicts a so called *tribar* which has been produced here by the author himself after the tribar published by R. Penrose in "British Journal of Psychology", Vol. 49, Part 1, February 1958.

Locative Temporal Logic and Distributed Real-Time Systems

—Specification—

een wetenschappelijke proeve op het gebied
van de Wiskunde en Informatica

Proefschrift

ter verkrijging van de graad van doctor
aan de Katholieke Universiteit Nijmegen,
volgens besluit van het College van Decanen
in het openbaar te verdedigen
op donderdag 13 oktober 1994,
des namiddags te 3.30 uur precies

door

Martin Josef Wieczorek

geboren op 22 april 1958 te Bochum (Duitsland)

PROMOTORES: PROF. DR. IR. J. VYTOPIL

PROF. DR. J.-J. CH. MEYER
(UNIVERSITEIT UTRECHT)

Für Claudia, Christina und Sarah

In memoriam Lisa

Similia similibus specificatur!
(Like specifies like!)

This principle has been adopted from the basic medical principle "Similia similibus curentur!" (Like cures like!).

The medical principle was pronounced first by Theophrast von Hohenheim (Paracelsus) about 1520 in "Über die Medizin".

It was rediscovered by Samuel Hahnemann and chosen as basic principle of Homœopathy in "Organon der rationellen Heilkunde", 1810.

MANUSCRIPTCOMMISSIE: PROF. DR. W.-P. DE ROEVER
(CHRISTIAN-ALBRECHTS-UNIVERSITÄT KIEL)

DR. J.J.M. HOOMAN
(TECHNISCHE UNIVERSITEIT EINDHOVEN)

DR. W. VAN DER HOEK
(UNIVERSITEIT UTRECHT)

Preface

MOTIVATION FOR THIS THESIS

From the early days on, *sequential systems*, possibly non-deterministic ones, have been built with the intention to apply some functional transformation on some input data to produce, if and when terminating, a certain result. An example might be a sorting algorithm where a particular list of items (input) is transformed into the sorted counterpart (output) of that list.

Opposed to sequential systems *reactive systems* have been introduced in the literature [39]. The significant difference between the two lies in the fact that sequential systems are always expected to terminate (although this is not always true) and to deliver some final result whereas reactive systems are expected not to terminate (although their lifetime will most often be bounded) but to maintain ongoing interaction with the environment. Controlling of nuclear power plants and geo-stationary satellites might serve as examples for reactive systems.

Another class of computing systems is termed *parallel systems*. There, not the question of termination is significant but the question of truly parallel or interleaved parallel execution of different tasks. Recently neural networks have appeared which can be considered an example of a (highly) parallel computing system.

Real-time systems, a further example of a class of computing systems, have the characteristic feature that the time of the occurrence of events or the time between two occurrences of events becomes important. Suppose a command has been given by the ground control station to prevent a satellite in the orbit from drifting away. Then, one would expect that the response of the satellite, for example the confirmation of position changing, will reach the ground within some short amount of time because otherwise the satellite could drift away.

Technological change and user needs for decentralized computing have forced people to build more and more complex systems. One class of systems play an important role in the every day life in our community because life itself has become extremely dependent on them. Take, for example, the control system needed for space shuttle missions including the ground control system as well as the shuttle control system. Or think of a traffic control system where unwanted chaotic states in cities or on highways would appear when they are failing. Regard, for example, the impacts of failing stop lights on the happenings at a crossing point. In our view, the properties of distribution and time dependency are relevant and common to all such systems for which reasons we will call them *distributed real-time systems*. Because of their complexity and their impact on every day life the quest for suitable models will be urgent.

In general, models for computing systems can be divided into two main classes: those having as basic assumption that executions can be totally ordered, i.e., states or transitions in an execution are elements of a totally ordered set. This approach is extensively used for sequential systems and has been adopted to reactive and parallel systems (see, for example, Manna and Pnueli [64]). The alternative approach rejects the assumption of total ordering and makes use of partial orderings (see, for example, Reisig [73]). The fundamental distinction between the two lies in the fact that the first one defines projections from the set of all occurrences of events, which are observable, onto a quite often linear time scale (time-observer-based models) whereas the second one uses a causal structure to order occurrences of events (cause-effect-based models).

The main disadvantage of the time-observer-based models, as provided in the literature by several authors, is that the observer is a sequential one also in the case of distributed real-time systems. In our view, it will be more intuitive to have a *distributed observer* in the case of distributed systems. Doing so it will lead to the possibility to differentiate in the specifications between *local* properties, i.e., properties local to some location—think of a processor or process—, and *distributed* properties, i.e., properties common to some or all such locations.

Hence, in this thesis, we will at first develop a *space and time model* for distributed real-time systems, that is, a model that is based on observations made by an *external observer in space and time.* Second, we will develop a *space and time logic* to reason about properties of distributed real-time systems, that is, a logic comprising a suitable language whose semantics is based on the space and time model and a deduction system providing basic properties and allowing us to derive other properties.

As a consequence of our space and time model and logic, the specifications of distributed real-time systems will be more structured than, e.g., corresponding temporal logic specifications thereby not neglecting the requirement of implementation bias freedom. We shall need less proposition and predicate symbols because the same property at different locations or time points asks for one symbol only. Thereby, note that we will confine ourselves, in this thesis, to specification issues and leave out any formal treatment of a programming language for distributed real-time systems.

OVERVIEW

This thesis is divided into two parts: part I provides the logical foundations and introduces our specification method. Part II discusses primitive communication techniques in the light of our specification method and presents specifications and verifications of two paradigms from the field of distributed real-time systems, i.e., atomic broadcast and group membership. Finally, appendix A provides a basic system of locative temporal logic (LTL) as it used in part II. In particular, this thesis proceeds as follows:

CHAPTER 1 Several distinct logics are applied in computer science for the specification and verification of system properties. They provide a rather low level of description language and most often stress only one aspect of reasoning thereby neglecting others. An introductory motivation will be given for our kind of logic, i.e., a logic that enables reasoning about temporal and locative properties in a unified formalism.

This chapter contains the following sections: Section 1.1 discusses classical logic-based specification and verification methods in the light of some general properties. A structural defect is derived. In section 1.2, the characteristic features of distributed real-time systems are discussed. In section 1.3, conclusions are drawn for a specification method based on modal logic that resolves the structural defect. Finally, section 1.4 introduces the paradigm of dining philosophers that is used throughout the first part of this thesis as a running example.

CHAPTER 2 Classical temporal logic has been shown valuable for reasoning about qualitative properties of various kinds of systems, e.g., transformational systems, message passing systems, and reactive systems. It has also been shown that the pure qualitative view on time will not be adequate for real-time systems. To resolve this deficiency metric temporal logic has been proposed: classical temporal operators have been extended by an index denoting the temporal distance between the involved time points.

This chapter contains the following sections: Section 2.1 provides a discussion of relevant properties of time and clocks concluding with a summary of those properties that we will attribute to our temporal reference space. In section 2.2, we will review metric temporal logic as introduced by Koymans [54]. Section 2.3 provides a temporal logic specification of the dining philosophers paradigm as introduced in the introduction.

CHAPTER 3 The idea of a locative logic has been pronounced by various people with different intentions at different times. Our motivation for a locative logic had been given by so called communication networks in distributed real-time systems. Communication networks can be regarded as graphs and as such they can be defined in terms of a set of locations together with a relation of reachability.

This chapter contains the following sections: Section 3.1 provides a discussion of relevant properties of space and nets concluding with a summary of those properties that we will attribute to our locative reference space. In section 3.2, we will introduce a basic version of locative logic with the additional feature of locative distances. Section 3.3 provides a locative logic specification of the dining philosophers paradigm as introduced in the introduction.

CHAPTER 4 The idea of having more than one dimension for reasoning about systems has been pronounced by various people with different intentions at different times. In all but one case, to our knowledge, these dimensions have been chosen to be equal, for example, two time dimensions. Our motivation for a locative temporal logic had been given by two aspects that are common to all distributed real-time systems: distribution of computing power and actions in real-time. Events in a distributed real-time system become dependent on the location and the time point of their occurrence. Thus a space-time model seems adequate for such systems.

This chapter contains the following sections: Section 4.1 discusses our locative temporal reference space. In section 4.2, we will formally define a basic version of locative temporal logic with the additional features of locative and temporal distances. Section 4.3 provides a locative temporal logic specification of the dining philosophers paradigm as introduced in the introduction. In section 4.4, we will summarize the results from part I and draw some conclusions emerging from the several specifications of the dining philosphers paradigm.

CHAPTER 5 In part I of this thesis, we have developed a locative temporal reference space together with a locative temporal logic for specifying and reasoning about properties of distributed real-time systems. Part II of this thesis will now be devoted to some applications from the field of such systems. Before doing so in subsequent chapters, we will at first formalize some fundamental concepts that have already been discussed informally in chapter 1.

This chapter contains the following sections: Section 5.1, gives a short introduction to this second part. Especially the examples will be set in relation. In section 5.2, we provide a formalization of the notion of a distributed real-time system and some related concepts. These notions will be illustrated by the paradigm of a distributed watchdog.

CHAPTER 6 According to the two-sorted nature of our approach, several communication techniques can be discriminated w.r.t. their impact mainly on space or mainly on time. In the literature, we have found w.r.t. locative discrimination *point-to-point-based* and *diffusion-based* communication techniques and w.r.t. temporal discrimination *synchronous* and *asynchronous* communication techniques.

This chapter contains the following sections: Section 6.1 investigates point-to-point-based and diffusion-based communication techniques and section 6.2 investigates synchronous and asynchronous communication techniques. In section 6.3, we draw some conclusions resulting from our investigations in this chapter.

CHAPTER 7 A basic problem in distributed real-time systems is that of information dissemination among processes when the system or parts of it are sensitive of failures. A well-known example to overcome particular failure situations is an atomic broadcast protocol that provides a communication service to prevent a distributed real-time system from failing under certain conditions.

This chapter contains the following sections: Section 7.1 provides an informal discussion of the atomic broadcast service. In section 7.2, we present a formal specification of the atomic broadcast service. In section 7.3, we present the specification of protocols tolerating only particular processor failures, i.e., processor fail-stop without any network partitions. This class of protocols will be proved to satisfy the atomic broadcast service. In section 7.4, we draw some conclusions resulting from our investigations in this chapter.

CHAPTER 8 Another basic problem in distributed real-time systems is that of reaching agreement on processor group membership, i.e., which processors are functioning correctly at a certain point in time. A well-known solution is a processor group membership protocol that guarantees the same view among the correct processors in the system despite the occurrence of processor startups or failures.

This chapter contains the following sections: Section 8.1 provides an informal discussion of the processor group membership service. In section 8.2, we present a formal specification of the processor group membership service. In section 8.3, we present a formal specification of a class of protocols based on periodic broadcast. This class of protocols will be proved to satisfy the group membership service. In section 8.4, we draw some conclusions resulting from our investigations in this chapter.

CHAPTER 9 This chapter summarizes our investigations presented in part I and part II of this thesis. In particular, the general properties (cf. chapter 1) that one should require of an adequate specification method will be examined in the context of our specification method LTL. Still open points within our logical system LTL will be reviewed and possible directions for future work will be discussed.

This chapter contains the following sections: Section 9.1, provides a summary of our investigations undertaken in this thesis and draws some conclusions. Section 9.2 briefly reviews still open points within LTL and introduces some possible items for future work.

APPENDIX A In general, a logical system consists of the *set of well-formed formulae* given through the language of the logic and the notion of *consequence*, that is, "a particular formula is derivable from a set of formulae in the corresponding logic". The logical system for LTL to be presented here has been used throughout the whole part II of this thesis.

This appendix contains the following sections: In section A.1, we will summarize the language of LTL resulting from the investigations in part I and used in part II. Section A.2 provides the formal semantics of basic logical operators of LTL. Section A.3 contains the definition of LTL consequence, that is, definitions of dual and other non-primitive operators, axioms, and rules. At the end we will state some properties derivable within LTL.

NOTES FOR READING

The similarities of chapters 2, 3, and 4 w.r.t. organization and content are desired and intended by the author. The reasons for our decision had been given by needs of comparability: differences and correspondencies between temporal logic, locative logic, and locative temporal logic could so become clearer and sharper. The same arguments apply to organization and content of chapters 7 and 8.

In the proofs of part II, references to particular formulae will be made. We shall make use of a referencing mechanism that is divided into the following types of formulae:

1. references to formulae that are local to the proof at issue; these are represented by the number of the corresponding formula followed by a point, e.g., "2.",

2. references to formulae that are local to the specification at issue; these are represented by the number of the corresponding formula surrounded by parentheses, e.g., "(3)",

3. references to formulae that are contained in our logical system LTL as provided in appendix A; these are represented by the number of the corresponding formula preceded by "LTL-" and surrounded by parentheses, e.g., "(LTL-10)", and

4. references to formulae that are derived in our logical system LTL as theorems, propositions, etc.; these are represented by the number of the corresponding theorem, proposition, etc., preceded by "THEOREM ", "PROPOSITION ", etc., e.g., "THEOREM 6.1".

A last note for reading should be allowed: in this thesis, we shall make use of some fundamental concepts of graph theory. For those details that are not provided in subsequent chapters we refer the interested reader to the corresponding literature, e.g., [36, 37, 86].

ACKNOWLEDGEMENTS

This thesis had probably not been written without support from many different people and institutions. At first, there are my promotores Jan Vytopil and John-Jules Meyer who provided to me a scientific environment where I could develop from a vague idea a better understanding of the problems around modelling and specifying distributed real-time systems. Above all I benefitted from Jan Vytopil's practical and John-Jules Meyer's theoretical insights. It was not always easy to really find "my bridge" between theory and practice and to find subsequently feasible ways from one side of the bridge to the other and vice versa. But many encouraging discussions have helped me to put my research in a form as it is presented now in this thesis. Moreover, I am greatly indebted to John-Jules Meyer for his thorough reading, detailed comments, and competent hints to ameliorate former versions of this thesis.

I am grateful to Jozef Hooman and Wiebe van der Hoek for carefully reading the manuscript and providing detailed suggestions for improvements.

Many thanks go to my two roommates during my stay in Nijmegen: in the early period it was Peter Coesmans. Not only a joint paper resulted from our collaboration but it was also Peter who helped me to learn the dutch language. It was a "learning by doing". Since Peter's departure from Nijmegen (summer 1991), I have had a very pleasant and fruitful collaboration with Wim Koole, my second roommate. Also in this case a joint paper resulted. Wim's competent comments were invaluable for the progress of both my research and, in particular, this thesis. He provided me with a lot of hints to get more insight into the basic concepts of modal logic.

I would like to thank Hanno Wupper for many fruitful discussions in the period from summer 1989 to autumn 1991. He pronounced to me the valuable idea of separating functional, temporal, and locative properties in the area of real-time systems. Moreover, he interested me for the research work in the group "Real-Timesystemen" at the University of Nijmegen. Without him I would not have started this research.

The research described here could not have been done without the financial support by the Commission of the European Community (CEC), the Dutch Technology Foundation (STW), the Dutch Computer Science Research Foundation (SION), the Dutch Organization for Scientific Research (NWO), and the University of Nijmegen. In particular, I participated in the following projects:

1. CEC: ESPRIT Basic Research Action 3096 (SPEC) "Formal Methods and Tools for the Development of Distributed and Real-Time Systems",

2. STW: "Fault Tolerance: Paradigms, Models, Logics, Construction" under grant number NWI88.1517 and 790-600-1517, and

3. SION: "A Specification Language for Reliable Real-Time Systems" under grant number 612-317-016.

Furthermore, I would like to thank all the members of those projects for their patience and interest in my research. Special thanks are due to Willem-Paul de Roever who provided me with the right support in critical situations during the SPEC-project. But also Dov Gabbay, Amir Pnueli, and Pierre Wolper helped me to find "my bridge". I enjoyed many constructive discussions with Jos Coenen and Jozef Hooman during workshops of the projects and during mutual visits in Eindhoven and Nijmegen. Also thanks to Hetty and Henk Schepers for their kind accomodation in their flat during a visit in Eindhoven and for interesting discussions in this period.

Within the fault-tolerance project, Mathai Joseph provided me with the opportunity to present to him my ideas on modelling and specifying distributed real-time systems. I am grateful for his willingness and constructive comments.

Two ad hoc working groups had been installed in this period of my research: the first was called "Werkgroep NS-Bereikbaarheid " where I had the opportunity to present and discuss logical aspects of my research. Especially, I could benefit from Wiebe van der Hoek's logical insights into modal logic and correspondence theory. The second working group was called "Flaviu Cristian Discussion Group". It was a platform for discussions about the work performed by Flaviu Cristian and his colleagues. This had a great influence on chapters 7 and 8. All members are thanked for many stimulating discussions.

It was very encouraging to find helpful and cooperative colleagues and friends at the University of Nijmegen and, in particular, in the group "Real-Timesystemen" of Jan Vytopil. Especially, I am grateful to Mirèse Halders-Willems who was always at the right time at the right place and helped me, among other things, to solve many of my problems with various administrations.

During the first few months of my stay in Nijmegen Kees Koster and his family allowed me to live in lodgings in their house for which I would like to thank them all at this place.

Last but not least, I would like to thank Hartmut Ehlich, supervisor of my master thesis at the University of Bochum and head of the computer center there, for his friendly cooperation and especially for allowing me to make use of the services of the computer center since my departure from Bochum in 1989.

Finally, I am greatly indebted to Claudia, Christina, and Sarah for their love, patience, and support. Special thanks are also due to my parents for their love and support.

Nijmegen, The Netherlands
Bochum, Federal Republic of Germany

June, 1994 M.J.W.

Contents

Part I

Foundations: LTL

CHAPTER 1

INTRODUCTION

OUTLINE

Several distinct logics are applied in computer science for the specification and verification of system properties. They provide a rather low level of description language and most often stress only one aspect of reasoning thereby neglecting others. An introductory motivation will be given for our kind of logic, i.e., a logic that enables reasoning about temporal and locative properties in a unified formalism.

This chapter contains the following sections: Section 1.1 discusses classical logic-based specification and verification methods in the light of some general properties. A structural defect is derived. In section 1.2, the characteristic features of distributed real-time systems are discussed. In section 1.3, conclusions are drawn for a specification method based on modal logic that resolves the structural defect. Finally, section 1.4 introduces the paradigm of dining philosophers that is used throughout the first part of this thesis as a running example.

1.1 SPECIFICATION METHODS

Quite often, if not always, the choice for a particular *specification method*, i.e., a specification language together with a method for proving system properties is a personal taste of its author or user. This will also be true in our case, of course. Nevertheless, there are some general properties that one should require of an *adequate* specification method, that is, a specification method where the descriptional complexity will be comparable[1] to the complexity of the corresponding system:

- ▷ *Formality* to ensure concise and unambiguous specifications and rigorous proofs enabling serious discussions about details.
- ▷ *Abstractness* to ensure that all irrelevant details (implementation bias) must not be specified.
- ▷ *Expressiveness* to ensure that all important properties of the application area can be specified.
- ▷ *Separation of concerns* [2] to support structured specifications enforcing a clear understanding of the necessary details.
- ▷ *Unity within diversity* [3] to establish a relationship between seemingly diverse objects within a unified framework.

The first three properties will be constituent for *what* adequacy is whereas the last two properties determine a possible way *how* to reach adequacy. Formality, abstractness, and expressiveness are obvious and have been referred to in the corresponding literature for many times (a more recent reference is [54] by Koymans). The last two properties—the really driving forces for the investigations undertaken in this thesis—ask for further clarification.

1.1.1 SEPARATION OF CONCERNS

Separation of concerns is a *structural property* and really means *a priori fixed* separation of concerns explicitly visible in the specification method. It highly depends on its application area (see section 1.2 for more details). It can be achieved in many different ways. Let us regard three logical approaches that

[1] Comparability between the two complexities has also been posed for transition systems by Husberg [45].

[2] The old paradigm of separation of concerns has been pronounced to us by Hanno Wupper as a valuable principle for the specification and verification of reliable real-time systems.

[3] This phrase is borrowed from Davis and Hersh [23].

are widely accepted as basis for the specification and verification of distributed real-time systems: pure first-order predicate logic as applied by, e.g., Lamport [56] and Cristian et al. [20], Hoare logic as applied by, e.g., Hoare [41], Zwiers [96], and Hooman [43], and temporal logic as applied by, e.g., Pnueli [69], Barringer et al. [5], and Manna and Pnueli [64].

In pure first-order predicate logic, a priori fixed separation of concerns is not at all present. Suitable means for reasoning have to be provided in the specifications itself. Moreover, a formal notion of program correctness will be missing, thus violating the formality requirement. On the other hand predicate logic gives much freedom to the user himself.

In classical Hoare logic, separation of concerns is manifested in a Hoare triple:

$$\{\varphi\}\ \pi\ \{\psi\}$$

where a program π is separated from its pre-condition φ and its post-condition ψ.[4] As it has been invented originally for reasoning about sequential programs [41], the structure of these formulae supports our thinking about such programs in an intuitive and convincing way. On the other hand classical Hoare logic will, in general, not be applicable to distributed real-time systems: it is a logic for specifying and proving partial correctness of programs, i.e., correctness in the case that the corresponding program terminates, but distributed real-time systems are most often built from reactive programs [39], i.e., programs that are supposed not to terminate. In this sense, classical Hoare logic violates, w.r.t. distributed real-time systems, the expressiveness requirement above. To get rid of such deficiencies extensions of classical Hoare logic have been suggested, e.g., by Hooman [43].

In a formalism based on classical temporal logic (SAT-formalism), separation of concerns is present in the following respects. *Firstly*, a separation similar to Hoare triples is used for the correctness formulae:

$$\pi \text{ SAT } \varphi$$

where π again is a program and φ is a specification written in temporal logic. *Secondly*, time has been made explicit in the specification language by providing special temporal operators to reason about the underlying temporal structure. The basic temporal operators are *always* and *eventually*, respectively:

$$\Box \qquad \Diamond$$

[4] Note that pre- and post-conditions are themselves specified in pure first-order predicate logic.

With such temporal operators properties can then be specificed *indefinitely w.r.t. time points.* Thus, primitive propositions and predicates can be freed from temporal aspects. *Thirdly*, it has been shown by Manna and Pnueli [63, 64] that a particular class of properties can be identified with a particular *canonical temporal formula.* For instance, the canonical temporal formulae of *safety* and *liveness* properties expressing that "nothing bad will happen" and "eventually something good must happen" will be, respectively:

$$\Box\,\varphi \qquad\qquad \Diamond\,\varphi$$

Response and *persistence* properties expressing that "infinitely often something good will happen" and "all but finitely many times something good will happen" is specified by the following canonical temporal formulae, respectively:

$$\Box\Diamond\,\varphi \qquad\qquad \Diamond\Box\,\varphi$$

Nevertheless, formalisms based on classical temporal logic also have their shortcomings as has been shown by Koymans [54]: classical temporal logic only provides for qualitative temporal reasoning but quantitative timing properties such as *bounded response* and *periodicity* will also be present. To get rid of these shortcomings extensions have already been suggested, e.g., by Koymans [54] and by Harel, Lichtenstein, and Pnueli [40].

1.1.2 UNITY WITHIN DIVERSITY

A priori diversity is achieved, for example, by the requirement of separation of concerns. As the latter requirement, unity within diversity also depends on its application area (see section 1.2 for more details). Unity within diversity is again a *structural property* and it is attainable in many different ways. Let us, once more, consider the three logical approaches as mentioned above, i.e., pure first-order predicate logic, classical Hoare logic, and classical temporal logic.

Pure first-order predicate logic is a very general framework that unifies in some sense other logics, e.g., modal logics—although we know from correspondence theory that there is no exact one-to-one correspondence between modal logic and first-order predicate logic [7]. Pure first-order predicate logic gives most freedom to its user but, on the other hand, no structural elements are a priori provided except for the usual distinction between individual variables and predicate and function symbols. It follows that unity within diversity has been achieved in predicate logic to a great extent but a priori fixed separation of concerns and thus diversity will be missing.

Classical Hoare logic unifies sequential reasoning within the diversity of pre- and post-conditions and programs.[5] This unity is logically manifested in Hoare triples $\{\varphi\}\ \pi\ \{\psi\}$ and the *rule of consequence*:

$$\frac{\varphi \rightarrow \varphi' \ ,\ \{\varphi'\}\ \pi\ \{\psi'\}\ ,\ \psi' \rightarrow \psi}{\{\varphi\}\ \pi\ \{\psi\}}$$

Also in this case it is obvious that logical derivation (deduction) is responsible for the unity within diversity.

In temporal logics, too, logical derivation (deduction) provides the means for unification. Time has been separated from other aspects such that primitive predicates now will be definable indefinite w.r.t. some time scale. Logical derivation, therefore, becomes temporal derivation, that is, reasoning about system properties is unified by temporal reasoning. In [54], Koymans describes this as "the specification method is based on a single formalism covering all aspects of a specification" and later on "the specifications remain purely temporal".

Going beyond the pure temporal framework we find in the theory of modal logic, of which temporal logic is a derivative, a foundation that gives rise to unity within *more* diversity, i.e., many-sorted modal logic. We will come back to this discussion in section 1.3 on modal logic. A kind of many-sorted modal logic has also been developed by Stuhlmann-Laeisz [83] (unfortunately, Stuhlmann-Laeisz called his logic himself many-dimensional). An analogous theory namely that of many-dimensional modal logic has been developed by Venema [87]. The two approaches differ in that many-sorted modal logic is founded on the direct product of possibly different universes whereas many-dimensional modal logic is founded on the direct product of equal universes (see Venema [87], footnote on page 46).[6]

1.1.3 A STRUCTURAL DEFECT

Common to classical logic-based specification and verification formalisms is the property that they allow for reasoning about the *global state* of a system, i.e., primitive propositions and predicates over particular domains, e.g., message and processor domains, will always be interpreted for the whole system. No distinction can be made explicit between *local* and *global* properties.[7] This is,

[5] By the way, this is also true for Dijkstra's weakest pre-condition logic [25]. The difference really lies in the analytical and synthetical character of Hoare and Dijkstra logic (cf. also [2]), respectively.

[6] From a programming language point of view (see, e.g., Algol 68 [92]), the difference is the same as between records and arrays.

[7] For classical temporal logic, this has already been observed by Pnueli in [70].

in our view, a considerable disadvantage or, as we call it, a *structural defect.* The *paradigm of an external observer in time* who investigates the behaviour of a system and tries to order the occurrences of events by projections onto one global time scale is adequate for sequential possibly non-deterministic systems but not for distributed real-time systems which are also built from truly parallel executing processes.[8]

The structural defect becomes obvious when we try to specify, e.g., termination in a distributed real-time system: should the termination time of the whole system be the *maximum of the individual termination times of the several processes*, should it be the *maximum at the different processors*, should it be the *set of the individual termination times of all processes*, or what criterium might it else be?

Regardless of a general solution a particular one for the termination problem might be to introduce as many propositional termination variables as individual components (processes or processors) are present in the system. Or instead of many propositional variables we can define one primitive predicate with a suitable parametrization over a domain of processes or processors. For example, let be given a predicate $terminated(t, p)$ indicating termination of a process p before or at time t. Then, the following formulae specify the maximum view of the overall termination time and the individual view, respectively:

$$\exists t : \forall p : terminated(t, p) \qquad\qquad \forall p : \exists t : terminated(t, p)$$

From first-order predicate logic, we also know that the first formula implies the second but not vice versa, i.e.,

$$[\ \exists t : \forall p : terminated(t, p)\ ] \rightarrow [\ \forall p : \exists t : terminated(t, p)\ ]$$

This coincides with our intuition that if we have a temporal upper bound such that each process has terminated before or at that point then for each process there exists a time point such that this process has terminated before or at that point. A third solution has been provided by Hooman [43] in that suitable assumptions or conditions, e.g., about no activity of components, are added in the corresponding proof rules. Our preferred solution has been suggested in [90] where we used a two-sorted modal logic.[9]

In general, there exist many solutions to cope with the problems resulting from the desirable distinction between local and global properties in distributed real-time systems. One class of solutions is founded in the *principle of*

[8] The same criticism has been pronounced by, among others, Reisig [73]. His preferred solution is causality-based, that is, not a total order semantics but a partial order semantics.

[9] Note that the term "two-sorted modal logic" has been suggested by Wim Koole [50].

compositionality of the corresponding semantics and proof system as indicated by de Roever [77] and later by Hooman [43]. In the latter case, for instance, a metric temporal logic[10] has been used together with a CSP-like programming language and a compositional verification method. Special predicates have been introduced in the temporal assertion language to reason about termination and communication.

For example, a constant predicate *done* has been used in [43] to express immediate termination without any further time delay. To cope with the problem of termination in parallel composition two different proof rules have been provided there: let π_1 and π_2 be two processes and let φ_1 and φ_2 be two assertions describing the properties of π_1 and π_2, respectively. If neither φ_1 nor φ_2 contains the termination predicate *done* then the rule of "simple parallel composition" [43] can be applied to get an assertion about the properties of the parallel composition of π_1 and π_2 from their corresponding assertions:

$$\frac{\pi_1 \text{ SAT } \varphi_1 \ , \ \pi_2 \text{ SAT } \varphi_2 \ , \ \text{neither } \varphi_1 \text{ nor } \varphi_2 \text{ contains } \mathit{done}}{\pi_1 \parallel \pi_2 \text{ SAT } \varphi_1 \wedge \varphi_2}$$

provided $dch(\varphi_i) \subseteq dch(\pi_i)(i = 1, 2)$ where *dch* is a function that delivers all directed (logical) channels referred to by a specification or program, respectively.

Such simple and intuitive rule will not be valid in the general case because in Hooman's maximal parallelism model $\pi_1 \parallel \pi_2$ will have terminated when both π_1 and π_2 have terminated. So one has to take into account that either π_1 terminates before π_2 or π_2 terminates before π_1 or π_1 and π_2 terminate at the same point in time. This is expressed by the disjunction in the rule of "general parallel composition" [43] where C and $\Box$ are the temporal operators "chop"[11] and "always", respectively, and *noact* is a predicate that is true when no activities take place on the corresponding channels, i.e., when all channels in the argument of *noact* are neither used for synchronization (waiting) nor for transmission:

$$\frac{\pi_1 \text{ SAT } \varphi_1 \ , \ \pi_2 \text{ SAT } \varphi_2}{\pi_1 \parallel \pi_2 \text{ SAT } (\varphi_1 \wedge (\varphi_2 \mathrm{C} \Box \mathit{noact}(dch(\pi_2)))) \vee (\varphi_2 \wedge (\varphi_1 \mathrm{C} \Box \mathit{noact}(dch(\pi_1))))}$$

Hence, if π_1 terminates before π_2 or both terminate at the same time point the parallel composition $\pi_1 \parallel \pi_2$ will satisfy $\varphi_2 \wedge (\varphi_1 \mathrm{C} \Box \mathit{noact}(dch(\pi_1)))$

[10] A thorough investigation of metric temporal logic can be found in [54].

[11] Informally speaking, $\varphi \mathrm{C} \psi$ holds on a model if that model can be decomposed into two consecutive (sub-) models such that φ holds on the first and ψ holds on the second (sub-) model. Sometimes, connective C is also called "combine".

expressing that the assertion about the parallel composition will be constructed from the component assertions φ_1 and φ_2 with the additional property that after termination of π_1 no activities will take place on the channels of π_1 (see right disjunct above). If π_2 terminates before π_1 or both terminate at the same time point the parallel composition $\pi_1 \parallel \pi_2$ will satisfy $\varphi_1 \wedge (\varphi_2 \mathrm{C} \Box \mathit{noact}(\mathit{dch}(\pi_2)))$ expressing that the assertion about the parallel composition will again be constructed from the component assertions φ_1 and φ_2 but now with the additional property that after termination of π_2 no activities will take place on the channels of π_2 (see left disjunct above). In case that π_1 and π_2 terminate at the same time point both disjuncts will be true, i.e., $\pi_1 \parallel \pi_2$ SAT $\varphi_1 \wedge \varphi_2 \wedge (\varphi_2 \mathrm{C} \Box \mathit{noact}(\mathit{dch}(\pi_2))) \wedge (\varphi_1 \mathrm{C} \Box \mathit{noact}(\mathit{dch}(\pi_1)))$. For more details, we refer the interested reader to Hooman [43]. We shall ourselves come back to this discussion in chapter 9 of this thesis.

An earlier result on composition of temporal logic specifications has been published by Barringer, Kuiper, and Pnueli [5] where the temporal formulae are interpreted over *labelled sequences of states.* This approach uses beyond global and local individual variables and state propositions so called *edge propositions* which allow to distinguish between transitions effected by a module and transitions performed by others. Hence, the structural defect from above is maintained although a distinction between transitions of different modules can be made explicit.

1.2 DISTRIBUTED REAL-TIME SYSTEMS

No commonly accepted definition exists of what a distributed real-time system exactly is or what at least properties are so that discrimination between a distributed real-time system and, e.g., a transformational system [39] makes sense. The following characterization of a distributed real-time system is inspired by Sloman and Kramer [82] and by Kopetz and Ochsenreiter [52]:

> *A distributed real-time system is a (computing) system consisting of several distinct processes which are possibly dispersed among spatially separated autonomous processors and which have to adhere to timing constraints. The processes coordinate their activities and interact in order to cooperate to achieve an overall goal.*

Hence, the general purpose of a distributed real-time system is "to achieve an overall goal". But this will also be true for, e.g., transformational systems. What makes the distinction significant between distributed real-time systems

and others must, therefore, be manifested in the processes' properties "dispersed among spatially separated autonomous processors", "have to adhere to timing constraints", "coordinate their activities", and "interact in order to cooperate".

Coordination of the activities between several processes is reached by means of synchronization. Cooperation is done by means of message transmission over a *communication network.*

Adherence to timing constraints is founded on a certain degree of accuracy in the knowledge of time. This knowledge of time allows, e.g., for time stamping messages and for measuring time periods between occurrences of events. The source of knowledge of time[12] is called a *clock.* A particular configuration of clocks in a distributed real-time system is called a *clock system.*

1.2.1 COMMUNICATION NETWORKS

A communication network consists of a number of spatially separated autonomous processors together with (physical) channels between them.[13] We can think of these channels as being cables, wires, radio, or what ever can be used to connect a pair of processors so that communication in some sense becomes possible. That part of a distributed real-time system which is responsible for transferring information between spatially separated processors will be called *communication system.* It makes use of the underlying communication network.

A communication network itself provides the lowest level in a distributed real-time system, that is, processors and physical channels that are sensitive to hardware failure. Most often additional software will be added to overcome failure situations and to prevent the whole system from failing in case of particular hardware faults.

Communication networks differ in the number of processors, the number of channels, and the way processors and channels are related to each other. Some sample topologies are provided in figure 1.1. As usual, we will represent a communication network by graph theoretical means.

In figure 1.1, picture (a) represents a *completely connected network.* That is, every node in the network is linked to every other node in the network. Such a configuration provides high message throughput and low transmission time delays because message transmission between multiple nodes can proceed end-to-end in parallel. As a result the communication software can be simple

[12]This term is borrowed from Levi and Agrawala [60].

[13]When no additional qualification is added to the term "channel" it will become clear from the context where it is used.

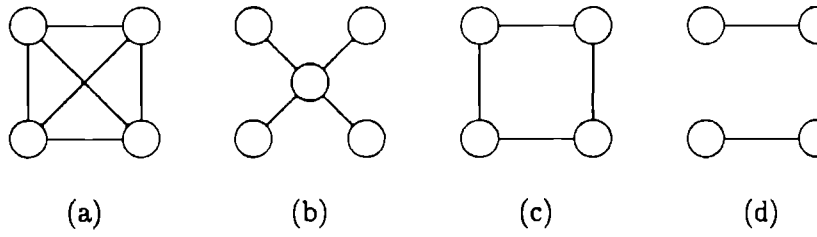


Figure 1.1: Sample Topologies of a Communication Network

because routing via intermediate nodes is not needed. Complete interconnections will also be of advantage when high reliability is demanded because multiple channels can be used for redundancy when channels or processors are failing. Opposed to such positive properties we have that the costs for complete interconnections will be high. Adding a new processor will result in n more channels, if n is the number of nodes in the old network, and some additional work will, in general, be necessary to manage the new channel at each node.

Picture (b) of figure 1.1 represents a *star network*. In this configuration, a central switching node is given to which all other nodes in the network are linked. All channels are single. Throughput of the network depends on the throughput of the central switching node. The transmission time delay depends at most on one intermediate node, namely the central one. A main disadvantage of this configuration type is its low reliability in the sense that if the central node fails then the whole network will suffer from it and communication no longer can take place.

Picture (c) of figure 1.1 represents a *ring network*. Expansion costs are low because only one new channel is needed when a new processor is added to the ring. Transmission time delays depend on the capabilities of each node. Most often it is assumed that communication is unidirectional and, hence, transmission will take place in directed rounds through the whole ring. The disadvantage of a ring lies in its low reliability because when a processor fails or a channel goes down communication in the whole network will be prevented.

Picture (d) of figure 1.1 represents a *disconnected network*. Such a configuration is of less interest because the two disconnected components cannot communicate with each other while for each such (connected) component the above discussion will apply.

1.2.2 CLOCK SYSTEMS

A clock system consists of a number of spatially separated clocks. Because most often the clocks are part of the corresponding processor it is usual to use the already installed communication network for clock controlling and reading. That part of a distributed real-time system which provides means for accessing and maintaining knowledge of time will be called here a *time system.*

Three basic types of clock systems (cf. figure 1.2) can be distinguished [60]: a *central clock system* as given in picture (a), a *master/slave clock system* as in picture (b), and a *distributed clock system* as presented in picture (c).

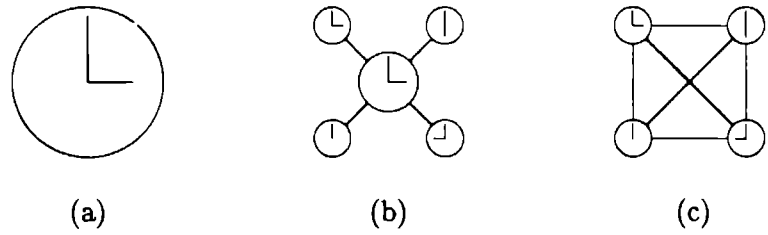


Figure 1.2: Sample Topologies of Clock Systems

In a central clock system, only one clock is available for the whole system. This clock is called the *central clock.* The time shown by this clock is provided to each process in the system. The communication traffic is low because at most one message is required for each process. Reliability w.r.t. time knowledge is low but it can be increased by a second, redundant *standby clock* which is used when the central clock fails. Problems due to synchronization do not occur because all processes refer to the same clock.

A master/slave clock system contains $n + 1$ clocks: one clock plays the role of a so called *master clock* which controls the n so called *slave clocks.* Getting the slave clocks synchronized with the master clock and keeping them synchronized is the main problem in such configurations. The communication traffic will not be much greater than in the previous type of clock system because the master has to provide the correction to the slave. What is more in the master/slave clock system is that the correction has to be applied by the slave to its local clock which will not be necessary in the central clock system because there is only one clock. The reliability of master/slave clock systems highly depends on the availability of the master clock: when the master fails a new master has to be elected. A failing slave clock only has impact on those processes with which it is associated but, in general, has no impact on the

whole system.

A distributed clock system contains n clocks of equal importance. The communication traffic is much higher than in the previous two systems because each site has to provide its own time value to all other sites in the system. A failing clock only has impact on the processes associated with it but, in general, has no impact on the whole system.

1.2.3 PROCESSES

A *process* in a distributed real-time system is an object with which behaviour can be associated. From a structural point of view, such a process consists of a *program* written in a certain programming language, a *processor* on which the program is to be executed, a *clock* for accessing and measuring time, and *channels* for coordination and cooperation with other processes. This view of a process is illustrated in figure 1.3.

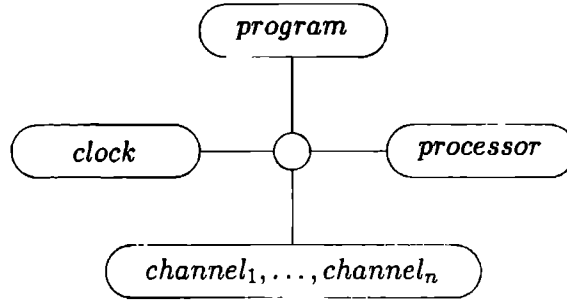


Figure 1.3: Process Components

A program here is understood as a possibly non-deterministic possibly communicating sequential program, that is, a program that may contain the usual constructs for variable assigment, sequential composition, deterministic or non-deterministic choice, and loops [93]. Moreover, because we have to do with coordination and cooperation between processes, constructs for synchronization and communication such as $c!e$ and $c?x$ [42] may also be contained in a certain program.[14]

[14]Note that it is not our goal in this thesis to provide a particular programming language. We only regard distributed real-time systems from a specification point of view. The interested reader is referred to, e.g., Hooman [43] for a recently published full account of a specification and verification method. In [90], we have ourselves suggested an extension of Hooman's approach by using our two-sorted modal logic.

Distribution of processes among processors may depend on many different aspects: for instance, the failure of processors may lead to a particular relation between processes and processors or, because of its task, a certain process must be associated with a special purpose processor. Another aspect is that of the kind of parallelism in a distributed real-time system. For example, do we want to allow for interleaved execution of processes or do we only allow for truly parallel execution? A comparison of the number of processes with the number of processors might be helpful: in case that the number of processes *is greater than* the number of processors there is *no one-to-one correspondence* between processes and processors. Therefore, it will not be possible that each process has its own processor and distribution of processes over processors will not be a simple task and must be determined by further conditions. In case that the number of processes *is less than* the number of processors there is again *no one-to-one correspondence* between processes and processors. But this time there are enough processors to allow each process to have its own processor and, therefore, distribution of processes over processors will be a simple task as long as no other conditions become relevant. Finally, in case that the number of processes *is equal to* the number of processors there is *a one-to-one correspondence* between processes and processors and, therefore, distribution of processes over processors will be as simple as in the second case.

Correctness of a distributed real-time system depends on the correctness the participating processes. Correctness of a participating process, in its turn, depends on the correctness of the process components. As the whole system may suffer from failing processes, a process itself may suffer from faulty components. From figure 1.3 we know that two different kinds of process components exist: software comprising programs and hardware comprising processors, clocks, and channels.

All process components will be sensitive to faults where *software faults* are "algorithmic faults stemming from unmastered complexity in the system design" [72].[15] *Hardware faults* may lead to a certain failure of a process or of the whole system although that process or system has been designed correctly. A particular hierarchy of failure classes that has been borrowed from Cristian et al. [20] has been provided in figure 1.4. There, an arrow indicates that the failure class at the start point is contained in the failure class at the end point.

A *timing failure* occurs when in response to a specified input the corresponding output is, in fact, provided but it is given too early (early timing failure) or it is given too late (late timing failure). The case where the cor-

[15]Note that in the remainder of this thesis we will not be concerned with software faults although it is very important even in the case of distributed real-time systems.

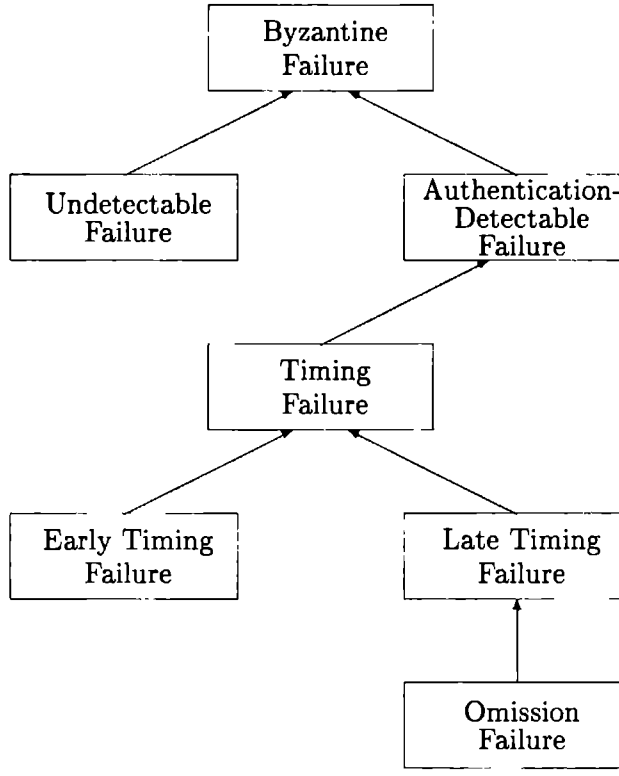


Figure 1.4: Failure Classification

responding output is never given (omission failure) can be regarded as a late timing failure with infinite arrival time. An *authentication-detectable failure* occurs when in response to a specified input an output is provided that is, in fact, the expected one but it is too early or too late or that is, at first sight, different from the expected one but whose defect can algorithmically be detected. The most general case is that of a *Byzantine failure.* It occurs when in response to a specified input an output is provided that is not the expected one but whose corruption is authentication-detectable or is undetectable.

For example, processors that have stopped their execution or processors that do not forward a message because of workload may cause omission failures. Also a transmission medium between processors that loses messages may cause omission failures. A clock that runs too fast or too slow so that time stamping of messages will not be correct may cause timing failures. Message

corruption by a faulty processor or by a faulty medium may cause Byzantine failures. If in case of Byzantine failures there exists a method for detecting and perhaps correcting a corrupted message, e.g., through a cyclic redundancy check (CRC) [79], then it is an authentication-detectable failure.

1.3 MODAL LOGIC

One of the origins of *modal logic* for the specification and verification of computer programs was layed, although less formal, in Burstall's paper on "Program Proving as Hand Simulation with a Little Induction" [12]. There, Burstall combines the method of Manna and Pnueli [62] and Floyd's method [27] to some kind of modal logic equivalent to S5. Burstall himself posed the question:

> "Further investigations, including a look at other modal operators like 'after', might be profitable, asking 'What kind of modal logic underlies our informal arguments about program executions?' "

The essence of modal logic is embodied in the concept of *possible worlds* or *sets of possible worlds.* As formalized by Kripke, a so called *Kripke model* comprises a set of possible worlds, a function assigning to each atomic formula, i.e., a primitive proposition a subset of possible worlds where it is true (*truth function*), and a relation between possible worlds providing those worlds that are accessible from a particular world (*accessibility* relation, *reachability* relation, *possibility* relation, *alternative* relation).

Depending on the application area the accessibility relation gives rise to different interpretations: in temporal logic, for example, this relation is defined on the set of time points and is interpreted as *temporal precedence* or *earlier* relation between points in time. Another example is that of dynamic logic where the relation is interpreted as *accessible through execution of a certain program.*

In this thesis, we will make use of another kind of logic that we will call *locative logic.* Besides the basic question of what a possible world denotes in a *locative structure* the most interesting question will be that about the interpretation of the accessibility relation. One half of the answer can be provided with the following citation of van Benthem [7]:

> "In our correspondence theory we also want to see the bare bones: a *frame* F is a couple (W, R) as above [W is the set of possible worlds and R is the accessibility relation], but without a valuation

> [truth function]. There is nothing intrinsically 'modal' about all this, of course. Frames are just the 'directed graphs' of Graph Theory."

The second half of the answer has been pronounced by Harary, Norman, and Cartwright [36]:

> "In interpreting point and line, the social scientist may be interested in relationships that indicate the possibility, the necessity, or the actual occurrence of an event. These different kinds of criteria are reflected by the words 'can', 'must', and 'does', that is, by the use of the potential, imperative, and indicative moods. Thus, a line uv may be interpreted as 'person u can communicate to person v', 'u must communicate to v', or 'u does communicate to v'. There is nothing in the nature of digraph theory which requires that the interpretation refer to any particular modality."

Together with our considerations in section 1.2 we may conclude: as interpretation for the possible worlds take the logical concept of a processor, that is, a location where something may happen and interpret the lines in the graph or, which is the same, the accessibility relation as "can communicate with". In such a way we get a Kripke frame of two sorts (cf. Koole [50]): a temporal structure to reason about temporal properties and a locative structure to reason about locative properties. Both structures will be static and do not evolve in time. This is obvious for the temporal structure but not for the locative one. That the locative structure is also static in time will be very important for our approach because graphs are most often used to model dynamically evolving networks, e.g., in [20] graphs may change in time and failure of a processor is modelled by constructing a subgraph in which the faulty processor is not a node. Dynamic evolution of networks in our approach must, therefore, be specified differently. This will be investigated in chapter 3.

Both structures together form our *reference space* for reasoning about properties of distributed real-time systems.[16] As the name indicates, a reference space is a conceptual means that must not be implemented but that allows to talk and reason about reality. In this sense, the reference space must also meet some properties. For the purposes in this thesis we will discuss the significant properties in chapter 2 for the temporal structure and in chapter 3 for the locative structure.

[16] Wittgenstein would call it *logical space* [94].

Of course, there exists some original work on logics dealing with operators for referencing to time and space, e.g., topological modal logics [74, 32, 83]. Such logics resulted from investigations in the context of natural languages where it is quite usual to refer to time points *and* places. E.g., the sentence "At 11:39:13.628 on the morning of Tuesday, January 28, 1986, the last bit of telemetry data was transmitted from the space shuttle Challenger."[17] contains the absolute time reference "At 11:39:13.628 on the morning of Tuesday, January 28, 1986" and the absolute space reference "space shuttle Challenger". Another use of locations can be found in [16]: the basic idea there was to let locations refer to (significant) *points in a system hierarchy* and to let the accessibility relation denote a kind of *subcomponent* relation.[18]

The differences between all such approaches are, in fact, founded in the underlying semantical concepts. What is thus new in this thesis is that we use locative logic to reason about communication networks.

1.4 A RUNNING EXAMPLE

At the end of this introduction, we start a discussion of a running example that will be used to illustrate the significant distinctions between the logics introduced in the subsequent chapters of part II and to demonstrate the adequacy of our approach of locative temporal logic. The running example is the well-known paradigm of "dining philosophers" that has been stated for the first time by Dijkstra in [24]. Our version is much simpler and is borrowed from Barringer et al. [6]. In this chapter, we will informally discuss the basic idea and provide a formal specification in first-order predicate logic.

1.4.1 THE PROBLEM

A number of philosophers want to dine together. In order to guarantee that every philosopher never goes hungry everyone has to eat sufficiently often. Because they are all civilized people they are provided with forks. The problem now is: on the one hand, the number of forks will be restricted so that not all philosophers can use forks at the same moment and, on the other hand, a philosopher will need two forks for eating. So, before a philosopher is allowed to eat he must gather the forks from his neighbours. Then, possessing one fork in his right hand and another fork in his left hand he can eat for at least two

[17] We have found this citation in Malcolm McConnel's book [66] about the Challenger disaster, page 244.

[18] The idea of locations being points in a system hierarchy and accessibility being a subcomponent relation originally stemmed from Hanno Wupper.

time units. In other words, having eaten needs forks for at least the previous two time units. Moreover, neighbours are not allowed to possess two forks at the same time (for more details on our version see [6]).[19]

EXAMPLE 1.1 (DINING PHILOSOPHERS) Suppose we have a community of four philosophers p_1, p_2, p_3, and p_4, say, and the same number of forks f_1, f_2, f_3, and f_4, say. To prevent the community from a hopeless muddle we assume that the philosophers will be sitting around a table and are only allowed to exchange forks with their neighbours. This is illustrated in figure 1.5.

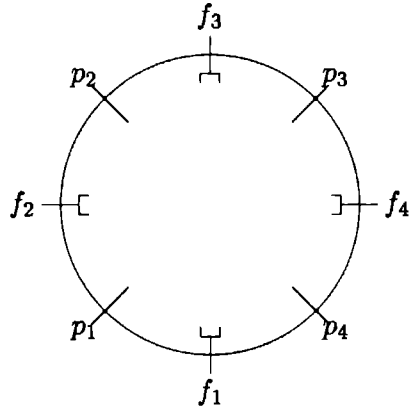


Figure 1.5: Dining Philosophers

For instance, philosopher p_3 is allowed to make use of forks f_3 and f_4 only. He will not be allowed to keep both forks infinitely long because then his two neighbours, i.e., p_2 and p_4 would never be able to eat. Moreover, philosopher p_3 himself will only be allowed to eat if he possessed two forks at the last two time points. ◇

1.4.2 THE MODEL

An axiomatization of our model of the dining philosophers community will comprise temporal properties as well as locative properties. Thereby, the locative properties characterize the structure of the community. Note that in this chapter we will make use of pure first-order predicate logic.

[19]Note that we will abstract from states such as thinking and being hungry (cf. Chandy and Misra [13]). For simplicity's sake, we only regard the state of having eaten.

TEMPORAL STRUCTURE We assume an infinite, discrete set T of time points together with a binary relation $<$ denoting *temporal precedence* between two time points, a binary relation $\neq_{\mathrm{T}}$ denoting *inequality* between two time points, and a function $|\cdot|_{\mathrm{T}}$ denoting the *distance* between two time points.

The properties of the temporal precedence relation are given as follows: (let, thereby, t, t', t'', and t''' be variables ranging over T)

1. $\forall t : \neg t < t$ (Irreflexivity)
2. $\forall t, t' : [\, t < t' \rightarrow \neg t' < t \,]$ (Asymmetry)
3. $\forall t, t', t'' : [\, t < t' \wedge t' < t'' \rightarrow t < t'' \,]$ (Transitivity)
4. $\forall t, t' : [\, t \neq_{\mathrm{T}} t' \rightarrow t < t' \vee t' < t \,]$ (Connectedness)
5. $\forall t, t' : [\, t < t' \rightarrow \exists t'' : [\, t < t'' \wedge \neg \exists t''' : t < t''' < t'' \,]\,]$
   $\wedge$
   $\forall t, t' : [\, t < t' \rightarrow \exists t'' : [\, t'' < t' \wedge \neg \exists t''' : t'' < t''' < t' \,]\,]$
   (Discreteness)
6. $\forall t : [\, \exists t' : t < t' \,]$ (No End)

LOCATIVE STRUCTURE We assume a finite set P of dining philosophers together with a binary relation $\approx$ denoting *locative reachability* between two philosophers, a binary relation $\neq_{\mathrm{L}}$ denoting *inequality* between two philosophers, and a function $|\cdot|_{\mathrm{L}}$ denoting the *distance* between two philosophers.

The properties of the locative reachability relation are given as follows: (let, thereby, p, p', p'', and p''' be variables ranging over P)

1. $\forall p : [\, p \approx p \,]$ (Reflexivity)
2. $\forall p, p' : [\, p \approx p' \rightarrow p' \approx p \,]$ (Symmetry)
3. $\forall p, p', p'' : [\, p \approx p' \wedge p' \approx p'' \rightarrow p \approx p'' \,]$ (Transitivity)
4. $\forall p : \exists p', p'' : [\, p \approx p' \wedge |(p, p')|_{\mathrm{L}} = 1 \wedge p \approx p'' \wedge |(p, p'')|_{\mathrm{L}} = 1 \wedge p' \neq_{\mathrm{L}} p'' \,]$
   (At Least Two Neighbours)
5. $\forall p, p', p'', p''' :$
   $[\, p \approx p' \wedge |(p, p')|_{\mathrm{L}} = 1 \wedge p \approx p'' \wedge |(p, p'')|_{\mathrm{L}} = 1 \wedge p \approx p''' \wedge |(p, p''')|_{\mathrm{L}} = 1$
   $\rightarrow$
   $p' = p'' \vee p' = p''' \vee p'' = p'''$
   $\,]$ (At Most Two Neighbours)

1.4.3 THE SPECIFICATION

For the specification of the requirements of the dining philosophers community we shall need additional predicates: (let, thereby, t, t', t'' and p, p', p'' be variables ranging over T and P, respectively)

- $eaten(p,t)$: philosopher p has just eaten at time t
- $lh(p,t)$: philosopher p possesses a fork in his left hand at time t
- $rh(p,t)$: philosopher p possesses a fork in his right hand at time t

Furthermore, let us introduce for notational convenience the following abbreviation for the possession of a fork in both hands:

$$bh(p,t) \stackrel{\text{def}}{=} lh(p,t) \wedge rh(p,t)$$

Finally, the community of dining philosophers has to satisfy the following requirements:[20]

EAT GUARANTEE All philosophers in the community will have eaten infinitely many times:

$$\forall p,p' : [\, p \approx p' \rightarrow \forall t : \exists t' : t < t' \wedge eaten(p',t') \,]$$

HAVING EATEN NEEDS FORKS A philosopher in the community can have eaten only if he possessed a fork in his left hand and a fork in his right hand at two successive points in time just before having eaten:

$$\begin{aligned} \forall p,p' : \forall t : \quad [\quad & p \approx p' \wedge eaten(p',t) \\ \rightarrow \quad & \\ & (\exists t' : t' < t \wedge |(t,t')|_{\mathrm{T}} = 1 \wedge bh(p',t')) \\ & \wedge \\ & (\exists t'' : t'' < t \wedge |(t,t'')|_{\mathrm{T}} = 2 \wedge bh(p',t'')) \\ ] & \end{aligned}$$

[20] Note that being a member of a particular community of dining philosophers will be modelled here by the locative reachability relation. As a consequence this means that *all* dining philosophers communities have to satisfy the corresponding properties or, in other words, to recognize a set of philosophers as a community of dining philosophers implies that all the properties must be valid for them.

EXCLUSION OF FORK POSSESSION If some philosopher possesses two forks then his neighbours may not possess two forks at the same point in time:

$$\forall p, p' : \forall t : [\ p \approx p' \wedge |(p, p')|_{\mathrm{L}} = 1 \rightarrow (bh(p, t) \rightarrow \neg bh(p', t))\ ]$$

Note that we have neglected any start condition for the dining philosophers community.

Chapter 2

Temporal Logic

Outline

Classical temporal logic has been shown valuable for reasoning about qualitative properties of various kinds of systems, e.g., transformational systems, message passing systems, and reactive systems. It has also been shown that the pure qualitative view on time will not be adequate for real-time systems. To resolve this deficiency metric temporal logic has been proposed: classical temporal operators have been extended by an index denoting the temporal distance between the involved time points.

This chapter contains the following sections: Section 2.1 provides a discussion of relevant properties of time and clocks concluding with a summary of those properties that we will attribute to our temporal reference space. In section 2.2, we will review metric temporal logic as introduced by Koymans [54]. Section 2.3 provides a temporal logic specification of the dining philosophers paradigm as introduced in the introduction.

2.1 THE TEMPORAL REFERENCE SPACE

As mentioned in the introduction, distributed real-time systems consist of a number of processes which have to adhere to timing constraints. Because processes do not have, in general, any common time reference—mainly because of spatial separation—coordination of activities and cooperation between processes will not be a simple task (cf. part II of this thesis). Uniqueness in time will a priori be missing.

Although *time is acausal* [1] we have decided to take time as one part of our reference space (cf. chapter 4). Having a temporal reference space at our disposal truth and falsity of propositions such as "message m has been received" will temporally be meaningful. Without it such propositions would be meaningless. Observe that the truth of such a proposition depends on the chosen reference point. A proposition whose truth or falsity is time-dependent will be called *indefinite w.r.t. time* or *temporally indefinite.*

Similarly a proposition such as "message m has been received at time t" will be meaningful when interpreted over a temporal reference space and t is an element of that space. In this case, of course, a temporal reference point is meaningless because the proposition itself contains an absolute time reference. A proposition whose truth or falsity is time-independent will be called *definite w.r.t. time* or *temporally definite.*

There is another pair of terms characterizing the type of temporal referencing, that is, referencing with a *pure future* fragment, referencing with a pure *past fragment*, or referencing with both a *past* and a *future* fragment. This choice is merely a question of personal taste: it has been shown that augmenting future time temporal logic with a bounded past fragment will not increase the expressive power of such a logic [61]. But as has also been demonstrated in the literature by several examples, having both past and future operators will lead to more natural and intuitive specifications than in the pure future case.

The most important kinds of properties of a temporal reference space will be the topological and metrical ones. These properties are directly induced by the application area, i.e., distributed real-time systems. For example, when chosing a discrete reference space but needing "continuous properties" the reference space will not be adequate. This is, in our view, also true for the opposite direction although Lamport argues that, e.g., a "discrete clock can be thought of as a continuous one in which there is an error of up to $\frac{1}{2}$ 'tick'

[1] Acausality of time, roughly speaking, means that time is not causally relevant to the occurrence of events [68]. In other words, a temporal order of events does not imply any causal order between them.

in reading it" [55]. Certainly, the reasons why the corresponding choices are not adequate, in our view, will be different (cf. chapter 1): in the first case, it is expressiveness and, in the second case, it is abstractness.

2.1.1 A THEORETICAL PERSPECTIVE

Topological and metrical properties can be associated with time. When following Swinburne [84] there will be no choice at all for such properties:

> "Time, being of logical necessity unique, one-dimensional and infinite, has of logical necessity a unique topology. Instants have to each other the neighbourhood relation of points on a line of infinite length."

Nevertheless, we will mention a few properties which we find relevant for distributed real-time systems from a theoretical point of view (for a detailed philosophical treatise on the structure of time see, e.g., Newton-Smith [68]).

MICRO-STRUCTURE

Two competing basic micro-structures are in use: *point structures* and *period structures*. Some recent research, e.g., Goswami, Bell, and Joseph [35], has shown the usefulness of period structures for real-time applications. Period structures have the advantage that they allow to represent a number of time points in a compact way without explicit quantification over those time points. Regard, for example, the following two formulae:[2]

$$(1) \qquad \forall i_1, i_2 \; : \; i_1 \; \mathsf{M} \; i_2 \wedge red(i_1) \wedge pressed(i_1) \rightarrow green(i_2)$$

$$(2) \qquad \forall t_1, t_2, t_3 \; : \; [\; t_1 \leq t_2 \leq t_3 \wedge (\forall t : t_1 \leq t \leq t_2 \rightarrow red(t)) \wedge (\exists t : t_1 \leq t \leq t_2 \wedge pressed(t)) \;] \rightarrow (\forall t : t_2 \leq t \leq t_3 \rightarrow green(t))$$

Both formulae express the property of a pedestrian light that "pressing

[2]Thereby, M is a relation between intervals expressing that the first interval is placed before the second one and that the right point of the first interval and the left point of the second one coincide.

the button during a *red phase* will later on lead to a *green* phase".[3] The first formula makes use of an underlying period structure whereas the second formula makes use of a point structure. It is obvious that the period-based formula is succinct and easier to read and understand than its point-based counterpart.

Obviously, there are also examples that demonstrate the same effect but with switched roles. As long as period structures are based on subsets of a linear point structure there will be no significant reason for one or the other structure except the personal taste. This is because point and period structures with properly chosen relations can be translated into each other [8].

A further aspect of micro-structure deals with the question of *denseness.* Three approaches have become prominent for specification issues: discrete time, dense time, and continuous time. Examples can be found in [53] by Koymans, Vytopil, and de Roever, in [43] by Hooman, and in [47] by Jackson, respectively.

Discrete time, e.g., the set $\mathbb{N}$ of natural numbers has the advantage that only one basic operator, i.e., "at the next moment"—for the past fragment it would be "at the previous moment"—must be introduced to express temporal properties (cf. Gabbay [28]). Dense time, e.g., the set $\mathbb{Q}$ of rational numbers will be more suitable when thinking of compositionality of the verification process of real-time systems (cf. Hooman [43]). Continuous time, e.g., the set $\mathbb{R}$ of real numbers will be interesting for reasons of mathematical convenience (cf. Lamport [55]).

A unifying variation of dense and discrete time structures has been provided by de Lemos, Saeed, and Anderson [59]. They introduce a so called *δ-dense time structure* $\langle T, <, \delta \rangle$ where $T \subseteq \mathbb{R}$ is a non-empty set of time points, $<$ is the precedence relation, and $\delta \in \mathbb{R}^0_+$ is the granularity, i.e., the distance between two adjacent time points. The precedence relation is assumed to be transitive, irreflexive, linear, and δ-dense which is the most interesting property there: (let t, t_1, t_2 be variables ranging over T)

$$\forall t_1, t_2 : [\, t_1 < t_2 - \delta \leftrightarrow \exists t : t_1 < t < t_2 \,]$$

It is now unifying in the following sense:

- $\delta = 0$ characterizes a dense time structure,
- $\delta > 0$ characterizes a discrete time structure with granularity δ, and

[3] Be aware of the fact that the two formulae alone do not strictly correspond to one another because, e.g., $red(i_1)$ translates to a universal quantification whereas $pressed(i_1)$ translates to an existential quantification. But this does not really affect our argument.

▷ $\delta = 1$ characterizes a subset of the integers as a special case of $\delta > 0$

CONNECTEDNESS

For us, the most important topological property of time will be that of connectedness. Most often it is assumed that time is open, i.e., open to the past and open to the future or at least open to the future. There might also be reasons for a closed topology (see picture (d) below) but this lies outside the scope of this thesis.[4] In figure 2.1, some sample topologies have been provided.

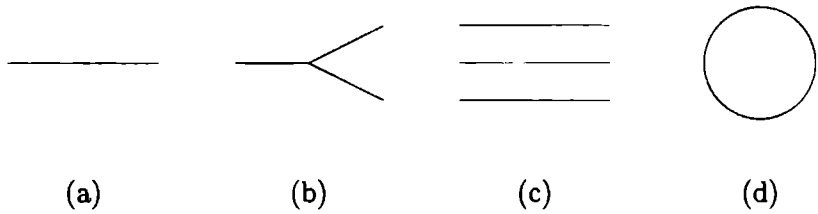


Figure 2.1: Sample Topologies

Picture (a) represents a linear flow of time open to both ends. This is also called the *standard topology of time.*[5] The set of all events will be totally ordered. Picture (b) contains the branching flow of time open to both ends. The future of a time point is always tree-like whereas the past is linear (for details on branching time temporal logic see, e.g., Ben Ari [10]). Any pair of events on different branches are temporally not related with each other. The third picture, i.e., picture (c) illustrates parallel time. In this case, events are temporally related to each other if they are associated with the same timeline otherwise they are temporally unrelated. In this sense, the set of all events will be divided by the temporal relation into equivalence classes where each such class contains all events lying on the same line.

METRICATION

Qualitative time structures are characterized by the fact that the basic time elements are related to one another by relations such as *before* or *after* (cf.

[4] For more details on closed structures see, e.g., Newton-Smith [68].

[5] Two metaphorical expressions denoting the standard topology of time are widely in use: *arrow of time* [51] and *river of time* [9].

McTaggart [67]) where topological properties are associated with such a relation (see above). Quantitative reasoning needs additional features in the time structures that allow to measure distances between the basic time elements.

Among the investigations on quantitative time structures we find two basic early approaches in the philosophical literature (for a detailed discussion see, for example, Rescher and Urquhart [75]): the first alternative is *additive time*. It makes use of a temporal group structure, i.e., a commutative group. The elements of the group are regarded to be time units and the group operation is addition between these time units. The specifications are made in terms of such time units relative to some origin. The origin itself will be given by the neutral element of the group.[6] The second alternative is *metric time* in the sense of Hausdorff [78]. The elements of the space are considered to be time points and in addition to the temporal precedence relation a distance function is used to measure the distance between two time points.

A further approach should be mentioned to overcome the deficiencies of a pure qualitative reasoning: While keeping the qualitative nature of time Pnueli and Harel [71] have added an *explicit global clock*. Fundamental to this approach is that it not only contains *rigid* variables but also *flexible* ones, i.e., variables whose value is fixed in the whole model and variables whose value may vary from time point to time point. An example of the former case is a variable denoting the value of a clock at a certain point in reference time (see below) and an example of the latter case is a variable denoting a particular clock.

2.1.2 A PRAGMATICAL PERSPECTIVE

Regarding time from a pragmatical point of view it is indispensable to include clocks. A clock itself can be regarded as the *source of knowledge of time* [60], that is, a device that can be used to refer to single points in time, e.g., time stamping events, or to measure time periods between occurrences of events. Properties of clocks and time are strongly related to each other. Following the standardization activities of time (cf. Andrewes [1]) there will be no choice at all for the properties:

> "[...] Therefore, in 1967, the atomic second was internationally adopted as the fundamental unit of time measurement, defined as

[6] This approach has been followed by the author himself [88] to define a three-sorted modal logic for the specification of fault-tolerant real-time systems. See also an early investigation by Garson [31] on the correspondence between relational and algebraical interpretations of modal formulae.

> the duration of 9,192,631,770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the cesium-133 atom. [...] The world's time signals (referred to as UTC—Universal Time Coordinated or Coordinated Universal Time) are coordinated at present by the Bureau International de Poids et Mesures in Paris. These signals, which are based on atomic time, are adjusted by leap signals in the middle or at the end of each year to keep them within nine-tenths of a second of the time determined by observations of the heavens (UT1—Universal Time 1)."

Nevertheless, we will mention a few properties which we find relevant for distributed real-time systems from a pragmatical point of view. Our synopsis is mainly based on the work by Kopetz [51], by Kopetz and Ochsenreiter [52], and by Lamport and Melliar-Smith [57].

CLOCKS AND TIME

Two kinds of clocks can be distinguished: A *physical clock*, that is, a hardware device which is based on some periodic physical oscillation (for instance, oscillation of a quartz crystal or a pendulum) in order to measure the progression of *physical time*. The period of oscillation is called the *granularity* of a physical clock. A *logical clock* is a software device which is based on some counter in order to measure the progression of *logical time* (see below).

Independent of being a physical or logical clock it can mathematically be represented as a function from one set of time points to another set of time points.[7] It is common practice to define such a function as a mapping C from a so called *reference time*[8] T_{ref} to a so called *clock time* T_{clock}, i.e., $C : T_{ref} \longrightarrow T_{clock}$. Another definition has been provided in [57] where domain and range of the above clock function are interchanged, i.e., $\widehat{C} : T_{clock} \longrightarrow T_{ref}$. Lamport and Melliar-Smith argue that for process control systems it is more appropriate to define a clock to be a mapping from clock time to reference time because then one can directly measure and compare occurrences of events in that reference time (for more details see [57]).

It may be sufficient to have infinite but discrete sets because physical clocks appear to be ticking in discrete steps, e.g., atomic second as cited above. However, for mathematical convenience, it will most often be assumed that

[7] We will also call such a function a *time-view function*. Note that this is a nice analogy with the space-view function in section 3.1.2.

[8] Sometimes it is also called *real time* [57, 18].

clocks are continuous functions rather than discrete ones. Lamport [55], for example, suggests to regard a discrete clock as a continuous one with a reading error of up to $\frac{1}{2}$ ticks. But independent of the choice for a discrete, dense, or continuous function it is important that the cardinality of T_{ref} is at least the cardinality of T_{clock}, i.e., $|T_{clock}| \leq |T_{ref}|$.

CORRECTNESS

The quality of the knowledge of time depends on the quality of its source, i.e., the correctness of the corresponding clock. Unfortunately physical clocks will not always be correct, that is, $C(t) = t$ not for all $t \in T_{ref}$ and, moreover, physical clocks may not even be monotonic, that is, if $t \leq t'$ then $C(t) \leq C(t')$ not for all $t, t' \in T_{ref}$. Hence, we would get quite often a picture like that in figure 2.2.

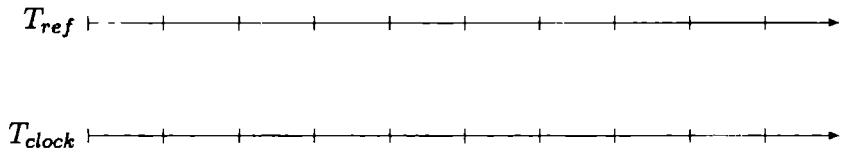


Figure 2.2: A Clock Sample

A clock must be presupposed to be at least monotonic because otherwise quite complicated situations may arise. For example, if a clock is not monotonic clock time may decrease from one point in reference time to a future point in reference time so that message time stamping delivers no suitable ordering.

The quality of the source of knowledge of time, i.e., the quality of a clock can thus be divided into three cases: a monotonic clock $C : T_{ref} \longrightarrow T_{clock}$ is called

- ▷ *correct at time* t iff $|C(t) - t| = 0$,
- ▷ ε*-accurate at time* t iff $|C(t) - t| \leq \varepsilon$, and
- ▷ *drifting at time* t otherwise.

Thereby, ε is an arbitrary but fixed upper bound on the amount of time that a clock may drift from the correct value during the execution of a distributed real-time system.

A monotonic clock C is called *correct* iff $|C(t) - t| = 0$ for all $t \in T_{ref}$ and it is called ε*-accurate* iff $|C(t) - t| \leq \varepsilon$ for all $t \in T_{ref}$.

EXAMPLE 2.1 Let be given some hardware clocks HC_1, HC_2, and HC_3. Hardware clocks may drift away from each other because of, e.g., hardware failure or different speeds.

Suppose that at a time point t_1 all clocks HC_1, HC_2, and HC_3 are correct, that is, all clocks satisfy the following conditions:

- $HC_1(t_1) = t_1$
- $HC_2(t_1) = t_1$
- $HC_3(t_1) = t_1$

This is illustrated by the first column in figure 2.3 (thereby, T denotes the temporal reference space and ϑ is some time distance).

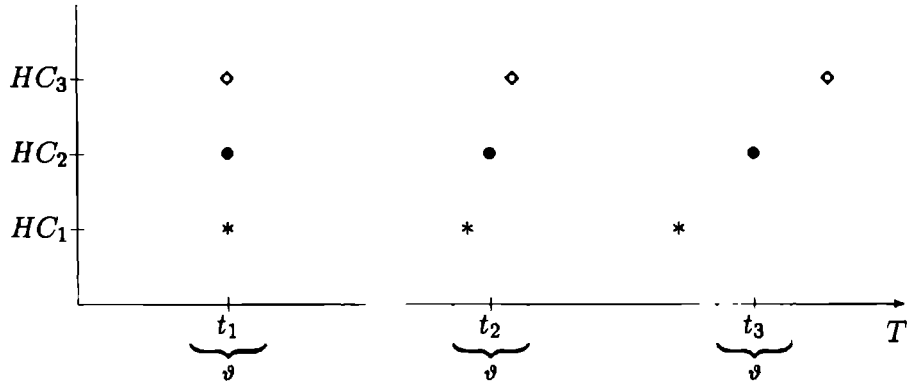


Figure 2.3: Drifting Distributed Clocks

Now suppose that clocks HC_1 and HC_3 are drifting away in such a way that HC_1 is at a speed a bit slower and HC_3 is at a speed a bit faster than the correct speed so that we get the following (qualitative and quantitative) clock conditions at time point t_2:

- $HC_1(t_2) < t_2$ and $t_2 - \frac{\vartheta}{2} < HC_1(t_2) < t_2 + \frac{\vartheta}{2}$
- $HC_2(t_2) = t_2$
- $HC_3(t_2) > t_2$ and $t_2 - \frac{\vartheta}{2} < HC_3(t_2) < t_2 + \frac{\vartheta}{2}$

Then, at time point t_2 clocks HC_1 and HC_3 will *not be correct* but within some a priori known bounds determined by a time constant ϑ and clock HC_2 is still correct (cf. figure 2.3).

Furthermore, suppose that at a time point t_3 clock HC_2 is still correct and that in the meantime no mechanism has prevented the clock system from drifting local clocks. Then, the following clock conditions would be valid at t_3 (cf. figure 2.3):

- $HC_1(t_3) < t_3 - \frac{\vartheta}{2}$
- $HC_2(t_3) = t_3$
- $HC_3(t_3) > t_3 + \frac{\vartheta}{2}$

It becomes clear that without any precautions a clock system where the distributed clocks are drifting away may be useless to distributed real-time systems because coordination of distributed activities or time stamping of messages, to give a few examples, cannot be guaranteed (although in [85] a different argumentation can be found). ◇

When correct physical clocks cannot be guaranteed for the execution time of a distributed real-time system it will be necessary to make the system tolerant against clock drifting of more than ε time units. This may happen, for example, because a physical clock is running too slow or to fast. To avoid such timing failures (cf. figure 1.4 in section 1.2) logical clocks will be introduced and maintained by so called *clock synchronization* algorithms (see [81] for an overview of clock synchronization). So what really is affected by such algorithms is rather a logical clock and not a physical one.

2.1.3 STANDARD TOPOLOGY WITH TEMPORAL DISTANCE

For the specification and verification of distributed real-time systems time as well as clocks will be needed. Time itself will be used twice: as an abstract notion on a meta-level to reason about events and as a concrete notion on an object-level as observable events of clocks. The meta-level time notion will be our temporal reference space or, as it would be called in modal logic, our temporal frame (see below). The object-level time notion will come into existence by so called fault-hypotheses on the corresponding clock system (for details see part II).[9]

Our temporal reference space will be constructed from the *standard topology of time* [68] together with a *temporal distance function* and a *temporal metric domain* [54]. Using first-order predicate logic with identity and a two-place predicate $R_T^{\leq} \subseteq T \times T$ we can express the properties of the temporal

[9]The distinction between meta-level time notion and object-level time notion has been inspired by Dov Gabbay's metabox concept [30].

reference space as follows: (let, thereby, t, t_1, t_2, and t_3 be variables ranging over the set T of time points)

1. $\forall t : \neg R^{\leq}_{\mathrm{T}}\, t\ t$ (Irreflexivity)

2. $\forall t_1, t_2 : R^{\leq}_{\mathrm{T}}\, t_1\ t_2 \rightarrow \neg R^{\leq}_{\mathrm{T}}\, t_2\ t_1$ (Asymmetry)

3. $\forall t_1, t_2, t_3 : R^{\leq}_{\mathrm{T}}\, t_1\ t_2 \wedge R^{\leq}_{\mathrm{T}}\, t_2\ t_3 \rightarrow R^{\leq}_{\mathrm{T}}\, t_1\ t_3$ (Transitivity)

   Asymmetry together with transitivity rule out the possibility that time has a closed structure (cf. picture (d) of figure 2.1) because if t_1 comes temporally before t_2 and tracing around the circle gives us by transitivity that t_2 comes temporally before t_1 which is a contradiction to asymmetry. Replacing asymmetry with irreflexivity leads to the same result.[10]

4. $\forall t_1, t_2 : t_1 \neq_{\mathrm{T}} t_2 \rightarrow R^{\leq}_{\mathrm{T}}\, t_1\ t_2 \vee R^{\leq}_{\mathrm{T}}\, t_2\ t_1$ (Connectedness)

   Irreflexivity together with transitivity and connectedness ensure that time is linear by ruling out non-linear structures such as given in pictures (b) and (c) of figure 2.1.

5. $\forall t_1, t_2 : t_1 \neq_{\mathrm{T}} t_2 \wedge R^{\leq}_{\mathrm{T}}\, t_1\ t_2 \rightarrow \exists t_3 : t_1 \neq_{\mathrm{T}} t_3 \wedge t_2 \neq_{\mathrm{T}} t_3 \wedge R^{\leq}_{\mathrm{T}}\, t_1\ t_3 \wedge R^{\leq}_{\mathrm{T}}\, t_3\ t_2$ (Denseness)

6. $\forall t_1 : \exists t_2 : R^{\leq}_{\mathrm{T}}\, t_1\ t_2$ (No End)

Opposed to the usual definition of a metric, i.e., a *real-valued* function, Koymans [54] has suggested to make the range of the distance function application-dependent but to require some minimal properties of the temporal metric domain. We will follow here Koymans and repeat the properties of the temporal distance function $\rho_{\mathrm{T}} : T \times T \longrightarrow \mathbb{D}_{\mathrm{T}}$ and the properties of its range $\mathbb{D}_{\mathrm{T}}$ using first-order predicate logic with identity and a two-place function symbol $+ : \mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}} \longrightarrow \mathbb{D}_{\mathrm{T}}$: (let, thereby, t_1, t_2, and t_3 be variables ranging over the set T of time points, let δ, δ_1, δ_2, and δ_3 be variables ranging over the set $\mathbb{D}_{\mathrm{T}}$ of temporal distances, and let $0 \in \mathbb{D}_{\mathrm{T}}$ be a constant temporal distance)

7. $\forall t_1, t_2 : \rho_{\mathrm{T}}(t_1, t_2) = 0 \leftrightarrow t_1 = t_2$ (Zero Distance)

[10] Because irreflexivity and transitivity together rule out closed structures we will, in the sequel, leave out asymmetry as a property of the temporal reference space. Moreover, asymmetry does not correspond to any modal formula (cf. Hughes and Cresswell [44], page 50).

8. $\forall t_1, t_2 : \rho_T(t_1, t_2) = \rho_T(t_2, t_1)$ (Symmetry)

9. $\forall t_1, t_2, t_3 : R^{\leq}_T\, t_1\ t_2 \wedge R^{\leq}_T\, t_2\ t_3 \rightarrow$
$\rho_T(t_1, t_3) = \rho_T(t_1, t_2) + \rho_T(t_2, t_3) \wedge \rho_T(t_3, t_1) = \rho_T(t_3, t_2) + \rho_T(t_2, t_1)$
(Conditional Equality)

10. $\forall \delta : \exists t_1, t_2 : \rho_T(t_1, t_2) = \delta$ (Surjectivity)
This ensures an adequate correspondence between the set T of time points and the temporal metric domain $\mathbb{D}_T$ of the temporal distance function.

11. $\forall \delta_1, \delta_2 : \delta_1 + \delta_2 = 0 \rightarrow (\delta_1 = 0 \wedge \delta_2 = 0)$ ($\mathbb{D}^0_T$)

12. $\forall \delta : \delta + 0 = \delta$ (Unit Element)

13. $\forall \delta_1, \delta_2, \delta_3 : (\delta_1 + \delta_2 = \delta_1 + \delta_3 \rightarrow \delta_2 = \delta_3)$ (Left Injectivity)

14. $\forall \delta_1, \delta_2, \delta_3 : (\delta_1 + \delta_3 = \delta_2 + \delta_3 \rightarrow \delta_1 = \delta_2)$ (Right Injectivity)

15. $\forall \delta_1, \delta_2, \delta_3 : (\delta_1 + \delta_2) + \delta_3 = \delta_1 + (\delta_2 + \delta_3)$ (Associativity)

16. $\forall \delta_1, \delta_2 : \delta_1 + \delta_2 = \delta_2 + \delta_1$ (Commutativity)

17. $\forall \delta_1, \delta_2 : \exists \delta : \delta_1 = \delta_2 + \delta \vee \delta_2 = \delta_1 + \delta$ (Absolute Difference)

2.2 METRIC TEMPORAL LOGIC REVISITED

For the specification of qualitative and quantitative properties of real-time (time critical) systems and for reasoning about them metric temporal logic has been introduced by Koymans [54].

In classical temporal logics (TL), temporal operators such as $\Box$ (always) and $\Diamond$ (eventually) restrict reasoning over the temporal universe by requiring a certain relationship between the time points at issue. If such a relationship does not exist nothing can be said about the existence of properties at those time points. Some qualitative extensions such as U (until) and S (since) [49] and $\triangle$ (everytime else) and $\triangledown$ (sometime else) [54, 76] have been added to enrich the expressive power of the propositional versions of classical temporal logics.[11] But such qualitative temporal operators will not suffice when quantitative properties come into existence.

[11]Note that Koymans and de Rijke use the symbols $\bar{D}$, meaning *all time points different from the actual one*, and D, meaning *some time point different from the actual one*, instead of $\triangle$ and $\triangledown$, respectively.

In metric temporal logic (MTL), restricting reasoning over the temporal universe has been resumed: indices have been added to the temporal operators to indicate the distance into past or future w.r.t. the actual temporal reference point. This makes it possible to specify properties relevant to real-time systems: for instance, two events can be related to one another by an a priori known time bound on the temporal distance between their occurrences.

2.2.1 LANGUAGE

The language of metric temporal logic as used in this chapter is a first-order language with identity where quantification over the metric domain $\mathbb{D}_T$ will be allowed. Opposed to that quantification over the set T of time points will not be allowed. The following special symbols and sets of symbols will be assumed:

- ▷ *individual constant symbols*: an enumerable set $\mathcal{A}_C$ of constant symbols containing a special symbol 0
- ▷ *individual variable symbols*: an enumerable set $\mathcal{A}_V$ of variable symbols
- ▷ *operation symbols*: infix function symbols $\oplus$, $=$, $<$ all of arity 2
- ▷ *propositional variable symbols*: an enumerable set $\mathcal{A}_P$ of constant predicate symbols
- ▷ *quantifiers*: $\forall$, $\exists$
- ▷ *propositional connectives*: $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
- ▷ *temporal connectives*: $\boxminus^T$, $\diamondminus^T$, $\boxplus^T$, $\diamondplus^T$, $\circledcirc^T$, $\circledcirc^T$, $\circledcirc^T$, $\circledcirc^T$, $\triangle^T$, ∇^T, $\circledcirc^T$, $\circledcirc^T$

  We shall indicate that a particular temporal connective is reflexive, i.e., refers to the reflexive closure of the underlying accessibility relation by writing a circle within the corresponding connective. A minus or plus sign within a connective indicates that it is a past or future connective, respectively.

The priorities of the propositional and temporal connectives are defined as usual in the following order (within one item the priorities are the same):

1. $\neg$, $\boxminus^T$, $\boxplus^T$, $\diamondminus^T$, $\diamondplus^T$, $\circledcirc^T$, $\circledcirc^T$, $\circledcirc^T$, $\circledcirc^T$, $\triangle^T$, ∇^T, $\circledcirc^T$, $\circledcirc^T$ (highest priority)
2. $\wedge$, $\vee$
3. $\rightarrow$, $\leftrightarrow$ (lowest priority)

The terms in metric temporal logic are used to denote elements of the temporal metric domain. Therefore, we will only need individual constant and variable symbols and the operation symbol:

DEFINITION 2.1 (WELL-FORMED TERMS) The set of well-formed terms in metric temporal logic is the minimal set of terms closed under the following formation rules:

1. An individual constant symbol is a well-formed term.
2. An individual variable symbol is a well-formed term.
3. If τ_1 and τ_2 are well-formed terms then $\tau_1 \oplus \tau_2$ is a well-formed term.

$\Diamond$

The last rule corresponds to the above mentioned need to add two distances from the temporal metric domain to get a new distance in that domain.

DEFINITION 2.2 (WELL-FORMED FORMULAE) The set of well-formed formulae in metric temporal logic is the minimal set of formulae closed under the following formation rules:

1. A propositional variable symbol is a well-formed formula.
2. The propositional connective $\bot$ (false) is a well-formed formula.
3. If φ_1 and φ_2 are well-formed formulae then $\varphi_1 \rightarrow \varphi_2$ is a well-formed formula.
4. If τ_1 and τ_2 are well-formed terms then $\tau_1 = \tau_2$ and $\tau_1 < \tau_2$ are well-formed formulae.
5. If τ is a well-formed term and φ is a well-formed formula then $\boxminus^{\mathrm{T}}_{=\tau}\ \varphi$, $\boxplus^{\mathrm{T}}_{=\tau}\ \varphi$, $\triangle^{\mathrm{T}}_{=\tau}\ \varphi$, and $\triangleq^{\mathrm{T}}_{=\tau}\ \varphi$ are well-formed formulae.
6. If δ is an individual variable symbol and φ is a well-formed formula then $\forall\delta : \varphi$ is a well-formed formula.

$\Diamond$

The language presented here is, in fact, a first-order extension of the metricated polymodal language $PML(R^{\mathrm{U}}_{\mathrm{T}}, R^{<}_{\mathrm{T}}, =_{\mathrm{T}}, \neq_{\mathrm{T}})$ with the universal relation $R^{\mathrm{U}}_{\mathrm{T}}$ on T, the temporal precedence relation $R^{<}_{\mathrm{T}}$ on T, the relation $=_{\mathrm{T}}$ of equality on T, and the relation $\neq_{\mathrm{T}}$ of inequality on T. Informally the basic temporal operators have the following meaning:

- ▷ *Always in the Past (at a certain distance)* ($\boxminus^{\mathrm{T}}_{=\tau}\ \varphi$) Formula φ is true at all points in time that precede the actual temporal reference point and whose distance to that point is equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- ▷ *Always in the Future (at a certain distance)* ($\boxplus^{\mathrm{T}}_{=\tau}\ \varphi$) Formula φ is true at all points in time that are preceded by the actual temporal reference point and whose distance to that point is equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- ▷ *Everytime but Different (at a certain distance)* ($\Delta^{\mathrm{T}}_{=\tau}\ \varphi$) Formula φ is true at all points in time that are different from the actual temporal reference point and whose distance to that point is equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- ▷ *Everytime (at a certain distance)* ($\triangleq^{\mathrm{T}}_{=\tau}\ \varphi$) Formula φ is true at all points in time whose distance to the actual temporal reference point is equal to the value of τ.

Non-primitive metric temporal operators, e.g., dual and reflexive versions of the above mentioned temporal operators can be defined in terms of the basic ones. We will give a list of the most important metric temporal operators in section 2.2.3 below.

2.2.2 FORMAL SEMANTICS

As usual we give a Kripke style semantics for our temporal language, that is, we first introduce the notions of a temporal frame, a metric temporal frame, and a metric temporal model and define afterwards the semantics of the formulae inductively on the complexity of their structure.

DEFINITION 2.3 (TEMPORAL FRAME) Let T be a non-empty set of elements (*time points*) and $R^{<}_{\mathrm{T}}$ be a binary relation on T (*temporal precedence*). Then, the structure $(T, R^{<}_{\mathrm{T}})$ is called a *temporal frame* or, for short, a *T-frame*. $\Diamond$

In the sequel, we will presuppose the properties of the standard topology with temporal distance as presented in section 2.1.3. Furthermore, we shall make no distinction between the alphabet (syntactic domain) $\mathcal{A}_C$ and the semantic domain $\mathbb{D}_{\mathrm{T}}$.

DEFINITION 2.4 (METRIC TEMPORAL FRAME) Let $\mathcal{F}_{\mathrm{T}} = (T, R^{<}_{\mathrm{T}})$ be a temporal frame and let $\mathbb{D}_{\mathrm{T}}$ be a non-empty set of elements (*temporal metric*

domain). Let $\rho_T : T \times T \longrightarrow \mathbb{D}_T$ be a $\mathbb{D}_T$-valued function (*temporal distance function*), $\oplus : \mathbb{D}_T \times \mathbb{D}_T \longrightarrow \mathbb{D}_T$ be a dyadic infix operator, and $0 \in \mathbb{D}_T$ be an element of the temporal metric domain (*unity element w.r.t.* $\oplus$). Then, the structure $(\mathcal{F}_T, \mathbb{D}_T, \rho_T, \oplus, 0)$ is called a *metric temporal frame* or, for short, a *metric T-frame.* ◇

DEFINITION 2.5 (EVALUATION OF TERMS) Let T be a set of time points and $t \in T$. Let Σ_T be the set of well-formed terms and $\alpha_T : T \times \Sigma_T \longrightarrow \mathbb{D}_T$ be a function assigning to each time point and well-formed term a distance. Then, α_T is defined inductively on the structure of terms in the following way:

$$\alpha_T(t,\tau) = \begin{cases} d & \text{if } \tau = d \text{ and } d \in \mathcal{A}_C \\ d & \text{if } \tau = \delta \text{ and } \delta \in \mathcal{A}_V \\ & \quad \text{and } \delta \in dom(\alpha_T(t)) \\ & \quad \text{and the value of } \delta \text{ is } d \text{ at time } t \\ \alpha_T(t,\tau_1) \oplus \alpha_T(t,\tau_2) & \text{if } \tau = \tau_1 \oplus \tau_2 \text{ is a compound term} \end{cases}$$

The function α_T is called a *temporal binding.* If α_T has the property that $\alpha_T(t,\tau) = \alpha_T(t',\tau)$ for all $t, t' \in T$ and all terms τ then it is called a *rigid temporal binding.* ◇

DEFINITION 2.6 (METRIC TEMPORAL MODEL) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{F}_T^{|\cdot|} = (T, R_T^{\leq}, \mathbb{D}_T, \rho_T, \oplus, 0)$ be a metric temporal frame. Furthermore, let α_T be a rigid temporal binding and $\mathcal{I}_T$ be a function assigning to each propositional variable $p \in \mathcal{A}_P$ a subset of T on which p is true (*temporal interpretation function*), i.e., $\mathcal{I}_T : \mathcal{A}_P \longrightarrow \wp(T)$. Then, the structure $(\mathcal{F}_T^{|\cdot|}, \alpha_T, \mathcal{I}_T)$ is called a *metric temporal model* or, for short, a *metric T-model.* ◇

DEFINITION 2.7 (TEMPORAL SATISFIABILITY) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{M}_T^{|\cdot|} = (\mathcal{F}_T^{|\cdot|}, \alpha_T, \mathcal{I}_T)$ be a metric temporal model and let $t \in T$. Then, a well-formed formula φ *holds in a metric temporal model* $\mathcal{M}_T^{|\cdot|}$ *at time point* t or φ *is satisfied in a metric temporal model* $\mathcal{M}_T^{|\cdot|}$ *at time point* t, notation $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \varphi$, is defined inductively as follows:

(1) $\mathcal{M}_T^{|\cdot|}, t \not\models_{TL} \bot$

(2) $\mathcal{M}_T^{|\cdot|}, t \models_{TL} p$ iff $t \in \mathcal{I}_T(p)$ for $p \in \mathcal{A}_P$

(3) $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \varphi_1 \rightarrow \varphi_2$ iff if $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \varphi_1$ then $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \varphi_2$

(4) $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \tau_1 = \tau_2$ iff $\alpha_T(t,\tau_1) = \alpha_T(t,\tau_2)$

(5) $\mathcal{M}_T^{|\cdot|}, t \models_{TL} \tau_1 < \tau_2$ iff $\alpha_T(t,\tau_1) < \alpha_T(t,\tau_2)$

(6) $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \boxminus_{=\tau}^{\mathrm{T}} \varphi$ iff $0 \neq \alpha_{\mathrm{T}}(t,\tau)$
and
for all $t' \in T$:
if $R_{\mathrm{T}}^{\leq}\ t'\ t$ and $\rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(t,\tau)$
then $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t' \models_{\mathrm{TL}} \varphi$

(7) $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \boxplus_{=\tau}^{\mathrm{T}} \varphi$ iff $0 \neq \alpha_{\mathrm{T}}(t,\tau)$
and
for all $t' \in T$:
if $R_{\mathrm{T}}^{\leq}\ t\ t'$ and $\rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(t,\tau)$
then $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t' \models_{\mathrm{TL}} \varphi$

(8) $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \triangle_{=\tau}^{\mathrm{T}} \varphi$ iff $0 \neq \alpha_{\mathrm{T}}(t,\tau)$
and
for all $t' \in T$:
if $t \neq_{\mathrm{T}} t'$ and $\rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(t,\tau)$
then $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t' \models_{\mathrm{TL}} \varphi$

(9) $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \triangleq_{=\tau}^{\mathrm{T}} \varphi$ iff for all $t' \in T$:
if $R_{\mathrm{T}}^{\mathrm{U}}\ t'\ t$ and $\rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(t,\tau)$
then $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t' \models_{\mathrm{TL}} \varphi$

(10) $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \forall \delta : \varphi$ iff for all α'_{T} different from α_{T} just on δ:
$\mathcal{M}_{\mathrm{T}}^{|\cdot|}[\alpha'_{\mathrm{T}}/\alpha_{\mathrm{T}}], t \models_{\mathrm{TL}} \varphi$

where $R_{\mathrm{T}}^{\mathrm{U}}$ is the universal relation $T \times T$. The model $\mathcal{M}_{\mathrm{T}}^{|\cdot|}[\alpha'_{\mathrm{T}}/\alpha_{\mathrm{T}}]$ results from the model $\mathcal{M}_{\mathrm{T}}^{|\cdot|}$ by replacing the binding α_{T} with the new binding α'_{T}, i.e., $\mathcal{M}_{\mathrm{T}}^{|\cdot|}[\alpha'_{\mathrm{T}}/\alpha_{\mathrm{T}}] = (\mathcal{F}_{\mathrm{T}}^{|\cdot|}, \alpha'_{\mathrm{T}}, \mathcal{I}_{\mathrm{T}})$.[12] ◇

DEFINITION 2.8 (TEMPORAL VALIDITY) Let $\mathcal{M}_{\mathrm{T}}^{|\cdot|} = (\mathcal{F}_{\mathrm{T}}^{|\cdot|}, \alpha_{\mathrm{T}}, \mathcal{I}_{\mathrm{T}})$ be a metric temporal model.

1. A well-formed formula φ *is valid in a metric temporal model* $\mathcal{M}_{\mathrm{T}}^{|\cdot|}$ or, for short, *T-valid in* $\mathcal{M}_{\mathrm{T}}^{|\cdot|}$, notation $\mathcal{M}_{\mathrm{T}}^{|\cdot|} \models_{\mathrm{TL}} \varphi$, iff for all $t \in T$: $\mathcal{M}_{\mathrm{T}}^{|\cdot|}, t \models_{\mathrm{TL}} \varphi$.

2. A well-formed formula φ *is valid*, notation $\models_{\mathrm{TL}} \varphi$, iff for all $\mathcal{M}_{\mathrm{T}}^{|\cdot|}$: $\mathcal{M}_{\mathrm{T}}^{|\cdot|} \models_{\mathrm{TL}} \varphi$.

◇

[12] Note that the universal relation $R_{\mathrm{T}}^{\mathrm{U}}$ has only been introduced for reasons of analogy between modal operators and relations on the corresponding universe

2.2.3 DEDUCTION

We will confine ourselves here to mention definitions, axioms, and rules that are basic to our metric temporal logic and that extend classical propositional calculi with properties for the temporal distance function and the temporal metric domain (see above for their statement in first-order predicate logic). Note that we are not aiming here at a complete axiomatization of metric temporal logic.

DEFINITIONS Most of the following definitions have been adopted from [54] but in our notation, e.g., ∇^{T} instead of D:

1. $\boxminus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta \rightarrow \boxminus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Past)
2. $\circledminus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : \circledminus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Past (refl.))
3. $\circledminus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} (\tau = 0 \wedge \varphi) \vee \boxminus^{\mathrm{T}}_{=\tau}\,\varphi$
4. $\diamondminus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \diamondminus^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Past)
5. $\diamondminus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\boxminus^{\mathrm{T}}_{=\tau}\,\neg\varphi$
6. $\circledcirc^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : \circledcirc^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Past (refl.))
7. $\circledcirc^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} (\tau = 0 \wedge \varphi) \vee \diamondminus^{\mathrm{T}}_{=\tau}\,\varphi$
8. $\boxplus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta \rightarrow \boxplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Future)
9. $\circledplus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : \circledplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Future (refl.))
10. $\circledplus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} (\tau = 0 \wedge \varphi) \vee \boxplus^{\mathrm{T}}_{=\tau}\,\varphi$
11. $\diamondplus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \diamondplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Future)
12. $\diamondplus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\boxplus^{\mathrm{T}}_{=\tau}\,\neg\varphi$
13. $\circledcirc^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : \circledcirc^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Future (refl.))
14. $\circledcirc^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} (\tau = 0 \wedge \varphi) \vee \diamondplus^{\mathrm{T}}_{=\tau}\,\varphi$
15. $\triangle^{\mathrm{T}}\,\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta \rightarrow \triangle^{\mathrm{T}}_{=\delta}\,\varphi$ (Everytime but Different)
16. $\nabla^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \nabla^{\mathrm{T}}_{=\delta}\,\varphi$ (Sometime but Different)
17. $\nabla^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\triangle^{\mathrm{T}}_{=\tau}\,\neg\varphi$
18. $\circledcirc^{\mathrm{T}}\,\varphi \stackrel{\text{def}}{=} \forall\delta : \circledcirc^{\mathrm{T}}_{=\delta}\,\varphi$ (Everytime)
19. $\circledcirc^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : \circledcirc^{\mathrm{T}}_{=\delta}\,\varphi$ (Sometime)
20. $\circledcirc^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\circledcirc^{\mathrm{T}}_{=\tau}\,\neg\varphi$

AXIOMS The axioms characterizing the temporal distance function and the temporal metric domain can be adopted from [54] into our language without any modifications apart from our notational conventions, e.g., $\mathbb{V}^{\mathrm{L}}$ instead of E:

1. $\vdash_{\mathrm{LTL}} \diamondplus^{\mathrm{T}} p \rightarrow \nabla^{\mathrm{T}} p$ (Irreflexivity)
2. $\vdash_{\mathrm{LTL}} \diamondplus^{\mathrm{T}} \diamondplus^{\mathrm{T}} p \rightarrow \diamondplus^{\mathrm{T}} p$ (Transitivity)
3. $\vdash_{\mathrm{LTL}} \nabla^{\mathrm{T}} p \rightarrow (\diamondminus^{\mathrm{T}} p \vee \diamondplus^{\mathrm{T}} p)$ (Connectedness)
4. $\vdash_{\mathrm{LTL}} \boxplus^{\mathrm{T}} p \rightarrow \diamondplus^{\mathrm{T}} p$ (No End)
5. $\vdash_{\mathrm{TL}} \boxplus^{\mathrm{T}} (\varphi_1 \rightarrow \varphi_2) \rightarrow (\boxplus^{\mathrm{T}} \varphi_1 \rightarrow \boxplus^{\mathrm{T}} \varphi_2)$ (F-Distributivity)
6. $\vdash_{\mathrm{TL}} \boxminus^{\mathrm{T}} (\varphi_1 \rightarrow \varphi_2) \rightarrow (\boxminus^{\mathrm{T}} \varphi_1 \rightarrow \boxminus^{\mathrm{T}} \varphi_2)$ (P-Distributivity)
7. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2 : \delta_1 \oplus \delta_2 = 0 \rightarrow (\delta_1 = 0 \wedge \delta_2 = 0)$ ($\mathbb{ID}_{\mathrm{T}}^0$)
8. $\vdash_{\mathrm{TL}} \forall \delta : \delta \oplus 0 = \delta$ (Unit Element)
9. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2, \delta_3 : (\delta_1 \oplus \delta_2 = \delta_1 \oplus \delta_3 \rightarrow \delta_2 = \delta_3)$ (Left Injectivity)
10. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2, \delta_3 : (\delta_1 \oplus \delta_3 = \delta_2 \oplus \delta_3 \rightarrow \delta_1 = \delta_2)$ (Right Injectivity)
11. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2, \delta_3 : (\delta_1 \oplus \delta_2) \oplus \delta_3 = \delta_1 \oplus (\delta_2 \oplus \delta_3)$ (Associativity)
12. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2 : \delta_1 \oplus \delta_2 = \delta_2 \oplus \delta_1$ (Commutativity)
13. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2 : \exists \delta : \delta_1 = \delta_2 \oplus \delta \vee \delta_2 = \delta_1 \oplus \delta$ (Absolute Difference)
14. $\vdash_{\mathrm{TL}} \forall \delta : \mathbb{V}^{\mathrm{T}} \mathbb{V}^{\mathrm{T}}_{=\delta} \top$ (Surjectivity)
15. $\vdash_{\mathrm{TL}} p \leftrightarrow \mathbb{V}^{\mathrm{T}}_{=0} p$ (Zero Distance)
16. $\vdash_{\mathrm{TL}} \forall \delta : [\ (p \wedge \mathbb{V}^{\mathrm{T}}_{=\delta} q) \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta} (q \wedge \mathbb{V}^{\mathrm{T}}_{=\delta} p)\ ]$ (Symmetry)
17. $\vdash_{\mathrm{TL}} \forall \delta_1, \delta_2 : [\ (\diamondplus^{\mathrm{T}}_{=\delta_1} \diamondplus^{\mathrm{T}}_{=\delta_2} p \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta_1 \oplus \delta_2} p) \wedge (\diamondminus^{\mathrm{T}}_{=\delta_1} \diamondminus^{\mathrm{T}}_{=\delta_2} p \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta_1 \oplus \delta_2} p)\ ]$ (Conditional Equality)

RULES The rules are given as usual in classical modal logics, i.e., modus ponens and necessitation. But note that two rules of necessitation will be needed here: one for future time and one for past time.

1. whenever $\vdash_{\mathrm{TL}} \varphi_1 \rightarrow \varphi_2$ and $\vdash_{\mathrm{TL}} \varphi_1$ then $\vdash_{\mathrm{TL}} \varphi_2$ (Modus Ponens)
2. whenever $\vdash_{\mathrm{TL}} \varphi$ then $\vdash_{\mathrm{TL}} \boxplus^{\mathrm{T}}_{=\tau} \varphi$ (F-Necessitation)
3. whenever $\vdash_{\mathrm{TL}} \varphi$ then $\vdash_{\mathrm{TL}} \boxminus^{\mathrm{T}}_{=\tau} \varphi$ (P-Necessitation)

For a discussion of soundness and (relative) completeness of the metric temporal deduction system and also for interesting properties that are derivable within the temporal calculus we refer the interested reader to Koymans [54]. But because we are mainly interested in the combination of the temporal and locative calculi we will avoid to mention such properties here. For the combined calculus, we refer to chapter 4.

2.3 A RUNNING EXAMPLE (CONTD.)

In the introduction, we have started a discussion about the well-known paradigm of dining philosophers at least for two reasons: firstly, we want to *illustrate* the principal distinctions between the various logics as introduced in this first part and, secondly, we want to *demonstrate* the adequacy of our specification method. Note that comparison and conclusions will be postponed till chapter 4 when all kinds of logics will have been defined.

Recall that we have provided in chapter 1 a specification of the dining philosophers in pure first-order predicate logic. In this chapter now, we will present a specification of those properties by making use of metric temporal logic as defined above.

2.3.1 THE MODEL

An axiomatization of our model of the dining philosophers community will again comprise temporal properties as well as locative properties. The properties of the temporal reference space have already been provided with the metric temporal logic above. Moreover, we will presuppose here a discrete time domain. What remains to be done in this section is a characterization of the locative structure.

LOCATIVE STRUCTURE We assume a set P of dining philosophers together with a binary relation $\approx \subseteq P \times P$ denoting *locative reachability* between two philosophers, a binary relation $\neq_{\mathrm{L}}$ denoting *inequality* between philosophers, and a function $|\cdot|_{\mathrm{L}}$ denoting the *distance* between two philosophers.

The properties of the locative reachability relation are given as follows: (let, thereby, p, p', p'', and p''' be variables ranging over P)

1. $\forall p : [\, p \approx p \,]$ (Reflexivity)
2. $\forall p, p' : [\, p \approx p' \rightarrow p' \approx p \,]$ (Symmetry)
3. $\forall p, p', p'' : [\, p \approx p' \wedge p' \approx p'' \rightarrow p \approx p'' \,]$ (Transitivity)

4. $\forall p : \exists p', p'' : [\ p \approx p' \wedge |(p,p')|_{\mathrm{L}} = 1 \wedge p \approx p'' \wedge |(p,p'')|_{\mathrm{L}} = 1 \wedge p' \neq_{\mathrm{L}} p''\ ]$
(At Least Two Neighbours)

5. $\forall p, p', p'', p''' :$
$[\ p \approx p' \wedge |(p,p')|_{\mathrm{L}} = 1 \wedge p \approx p'' \wedge |(p,p'')|_{\mathrm{L}} = 1 \wedge p \approx p''' \wedge |(p,p''')|_{\mathrm{L}} = 1$
$\rightarrow$
$p' = p'' \vee p' = p''' \vee p'' = p'''$
$]$ (At Most Two Neighbours)

2.3.2 THE SPECIFICATION

For the specification of the requirements of the dining philosophers community we shall need additional predicates:[13] (let, thereby, p and p' be variables ranging over P)

- $eaten(p)$: philosopher p has just eaten
- $lh(p)$: philosopher p possesses a fork in his left hand
- $rh(p)$: philosopher p possesses a fork in his right hand

Furthermore, let us introduce for notational convenience the abbreviation $bh(p)$ for the possession of a fork in both hands by philosopher p:

$$bh(p) \stackrel{\text{def}}{=} lh(p) \wedge rh(p)$$

Finally, the community of dining philosphers has to satisfy the following requirements:[14]

EAT GUARANTEE All philosophers in the community will have eaten infinitely many times:

$$\forall p, p' : [\ p \approx p' \rightarrow \boxplus^{\mathrm{T}} \diamondplus^{\mathrm{T}} eaten(p')\ ]$$

HAVING EATEN NEEDS FORKS A philosopher in the community can have eaten only if he possessed a fork in his left hand and a fork in his right hand at two successive points in time just before having eaten:

$$\forall p, p' : p \approx p' \rightarrow \boxplus^{\mathrm{T}} [\ eaten(p') \rightarrow \diamondminus^{\mathrm{T}}_{=1}\ (bh(p') \wedge \diamondminus^{\mathrm{T}}_{=1}\ bh(p'))\ ]$$

[13] Note that the predicates will now be indefinite w.r.t. time (cf. section 2.1).

[14] Note again that being a member of a particular community of dining philosophers will be modelled here by the locative reachability relation. As a consequence this means that *all* dining philosophers communities have to satisfy the corresponding properties or, in other words, to recognize a set of philosophers as a community of dining philosophers implies that all the properties must be valid for them.

EXCLUSION OF FORK POSSESSION If some philosopher possesses two forks then his neighbours may not possess two forks at the same point in time:

$$\forall p, p' : [\ p \approx p' \wedge |(p, p')|_L = 1 \rightarrow \boxplus^T (bh(p) \rightarrow \neg bh(p'))\ ]$$

Note that we have again (cf. chapter 1) avoided to mention any start condition for the dining philosophers community.

CHAPTER 3

LOCATIVE LOGIC

OUTLINE

The idea of a locative logic has been pronounced by various people with different intentions at different times. Our motivation for a locative logic had been given by so called communication networks in distributed real-time systems. Communication networks can be regarded as graphs and as such they can be defined in terms of a set of locations together with a relation of reachability.

This chapter contains the following sections: Section 3.1 provides a discussion of relevant properties of space and networks concluding with a summary of those properties that we will attribute to our locative reference space. In section 3.2, we will introduce a basic version of locative logic with the additional feature of locative distances. Section 3.3 provides a locative logic specification of the dining philosophers paradigm as introduced in the introduction.

3.1 THE LOCATIVE REFERENCE SPACE

As mentioned in the introduction, distributed real-time systems consist of a number of processes which are spatially separated among various processors. Processes may get a partial but coherent or a complete but incoherent view of the system. Therefore, processes have to coordinate their activities and have to cooperate in order to exchange information between one another. Coordination and cooperation will not be a simple task (cf. part II of this thesis). Uniqueness in space will a priori be missing.

Although *space is acausal* [1] we have decided to take some kind of space as one part of our reference space (cf. chapter 4).[2] Having a locative reference space at our disposal truth and falsity of propositions such as "message m has been received" will locatively be meaningful. Without it such propositions would be meaningless. Observe that the truth of such a proposition depends on the chosen reference point. A proposition whose truth or falsity is space-dependent will be called *indefinite w.r.t. space* or *locatively indefinite.*

Similarly a proposition such as "message m has been received at location l" will be meaningful when interpreted over a locative reference space and l is an element of that space. In this case, of course, a locative reference point is meaningless because the proposition itself contains an absolute space reference. A proposition whose truth or falsity is space-independent will be called *definite w.r.t. space* or *locatively definite.*

We ask again for other pairs of terms characterizing the type of locative referencing. Similar to the temporal division into *future* and *past* fragments (see chapter 2) we find the following locative fragments interesting: referencing with a pure *front fragment*, that is, the set of locations which are reachable from the actual location, referencing with a pure *back fragment*, that is, the set of locations from which the actual location is reachable, and referencing with both front and back fragments. For the same reasons as in temporal logic the choice for the one or the other case is merely a question of personal taste. Having both front and back operators will lead, in some cases, to more natural and intuitive specifications than in the pure front case (see also section 9.2).

The most important kinds of properties of a locative reference space will

[1] Acausality of space, roughly speaking, means that space is not causally relevant to the occurrence of events [68]. In other words, a spatial order of events does not imply any causal order between them.

[2] Note that the term *space* will be used here with two different meanings: firstly, space is a general concept similar to time and, secondly, space denotes a particular mathematical structure. In both senses, it must not be confounded with the four-dimensional space of modern physics and philosophy. The last case is mostly called *spacetime.*

be the topological and metrical ones (cf. Koole [50]). These properties must be adequately chosen and will be directly induced by the application area, i.e., distributed real-time systems. Recall from chapter 1 that adequacy here is determined by the notions of abstractness and expressiveness.

3.1.1 A THEORETICAL PERSPECTIVE

Topological and metrical properties can be associated with space. Analogous to our "temporal discourse" in chapter 2 we start our "locative discourse" with the same problem:

> Does space, being of logical necessity unique, three-dimensional and infinite, have of logical necessity a unique topology?

Opposed to his positive answer w.r.t. time Swinburne [84] does not argue the same way for space. It seems, when following Swinburne, that time and space are completely different in their nature.

So, let us discuss a few properties which we find relevant for distributed real-time systems from a theoretical point of view. Thereby, we will try to draw some connections to corresponding temporal properties.

MICRO-STRUCTURE

Let us mention here two different approaches that seem interesting and intuitive and that are both based on point structures: the first is founded on *graph-theoretical* the second on *geometrical* means. The first approach has been influenced by the strong resemblance between directed graphs and frames (cf. section 1.3). It follows the lines of classical one-dimensional temporal frameworks, that is, a universe of locations together with a binary relation containing ordered pairs of locations where the second location is reachable from the first location. Reachability in this sense provides for the static means to reason about communication between locations in a distributed system, that is, if a location l_1 wants to communicate with location l_2 then l_2 must be statically reachable from l_1. This approach is the preferred one in this thesis for which reasons we shall postpone any further discussion of details to subsequent sections.

The second approach makes use of real-space as a subspace of four-dimensional spacetime. In [3, 4], for instance, Baeten and Bergstra suggest a process algebra based on four-dimensional spacetime where three-tuples—as elements of $\mathbb{R}^3$—and time points—as elements of $\mathbb{R}$—are used to denote positions of processes in spacetime. Suppose, for example, that data has to be transmitted from a sender S to a receiver R via an intermediate transmitter T.

Thereby, both sender and receiver reside *fixed* at locations l^{S} and l^{R}, respectively, whereas the transmitter is *mobile*, that is, T is moving on a line between location l_1^{T} and location l_2^{T} such that its position changes in time. The transmitter's location l^{T} at time t, denoted $l^{\mathrm{T}}(t)$, is determined by the following formula:

$$l^{\mathrm{T}}(t) = l_1^{\mathrm{T}} + \frac{1 - \cos(\omega t)}{2} \cdot (l_2^{\mathrm{T}} - l_1^{\mathrm{T}})$$

where l_1^{T} is also chosen as the starting point of T at time $t = 0$. For more details, the interested reader is referred to example 4.6 in [4].

Work by Goldblatt [34] has shown that the modal sentences valid in the structure of four-dimensional spacetime are exactly the theorems of the logic S4.2. Four-dimensional spacetime has been defined in [34] as a structure $(\mathbb{R}^4, \leq)$ where $\mathbb{R}^4$ is the set of all real four-tuples and for $x = (x_1, \ldots, x_4)$ and $y = (y_1, \ldots, y_4)$ in $\mathbb{R}^4$ a relation $\leq$ is defined as follows:

$$x \leq y \text{ iff } \sum_{i=1}^{4-1} (y_i - x_i)^2 \leq (y_4 - x_4)^2 \text{ and } x_4 \leq y_4$$

The intended interpretation of $x \leq y$ is that "a signal can be sent from x to y at a speed at most that of the speed of light" so that y lies in the "causal future" of x. The relation $\leq$ is reflexive and delivers a partially-ordered and directed frame. For details, we refer the interested reader to [34] where also the case of signals slower than light is regarded.

A further aspect of micro-structure—with which we have been concerned also in the "temporal discourse" of chapter 2—is *denseness*. In the light of the above mentioned two approaches it is obvious that discreteness of the locative reference space should be required for the graph-theoretical approach whereas the spacetime approach leaves more possibilities, e.g., continuous spacetime or discrete spacetime. But investigations in these directions would go far beyond the scope of this thesis for which reasons we will finish our discussion on micro-structure with an open problem stated by Goldblatt (see item 2 on page 235 in [34]):

> "Analyse the logic of *discrete* spacetime (i.e. when $\mathbb{R}$ is replaced by $\mathbb{Z}$)."

CONNECTEDNESS

The most important topological property of space will be that of connectedness. Presupposing that we are interested to model space by graph-theoretical

means we can distinguish three classes of directed graphs:[3] *completely connected*, *strongly connected*, and *disconnected*. In figure 3.1, some sample topologies have been provided.

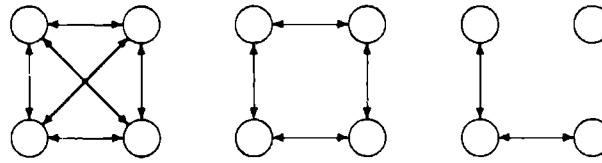


Figure 3.1: Sample Topologies

Picture (a) represents a completely connected directed graph: all locations are linked by a line, i.e., between every two locations there is an arrow from one to the other and vice versa. Exchange of information between locations can directly take place without any intermediate location. Picture (b) contains a strongly connected directed graph or, to be more precise, a ring: all locations are reachable from one another but not all locations are linked by a line as in picture (a). Exchange of information between locations possibly needs intermediate locations. The third picture, i.e., picture (c) represents a disconnected graph: there may exist a location which is isolated, i.e., between this location and all other locations there does not exist any line. Exchange of information with such isolated locations cannot at all take place. Note that disconnection in space has its temporal analogue in parallel time (cf. section 2.1.1) because events associated with different independent timelines are equally unrelated as events occuring at disconnected locations.

In the first two cases, i.e., pictures (a) and (b) in figure 3.1 we have the situation that the reachability relation on the set of locations is an equivalence relation. When we confine ourselves only to connected components of directed graphs then the reachability relation in the third case, i.e., picture (c) above constitutes an equivalence relation, too.

[3]Observe that we will take advantage of the fact that an undirected graph is a directed one with the additional property that if there is a line from one location to another then there is also a line from the latter to the former.

METRICATION

Qualitative space structures are characterized by the fact that the basic space elements are related to one another by relations such as *in front of* or *in back of* where topological properties are associated with such relations (see above).[4] Quantitative reasoning needs additional features in the space structures that allow to measure distances between the basic space elements (quantitative structures). Because the foundations for our locative logic are given by graph-theoretical means we will confine ourselves here to the discussion of distances within the framework of directed graphs.[5] Also in case of directed graphs the interpretation of distances may vary from application to application. For instance, a value providing the distance between locations in kilometers might be interesting for distributed real-time systems. For details on different possible interpretations of the distance concept in directed graphs the interested reader is referred to, e.g., Harary et al. [36].

So let L be a set of *locations* [6], E be a set of *directed edges*, and I be an *incidence function* associating with each directed edge an ordered pair of locations. Then the structure (L, E, I) is called a *directed graph*. The elements of the range of I will be called *arrows* here and will be denoted by $\langle l_1, l_2 \rangle$ if l_1 and l_2 are locations. A directed graph is *completely connected* if between every two locations in that graph there exists an arrow. Let l_1 and l_k be two locations in a directed graph. A *path from* l_1 *to* l_k is a sequence of arrows of the form $(\langle l_1, l_2 \rangle, \langle l_2, l_3 \rangle, \ldots, \langle l_{k-1}, l_k \rangle)$. Two locations in a directed graph are *mutually reachable* if there exists a path from the one to the other and vice versa. A directed graph in which every two locations are mutually reachable is *strongly connected*. The *length of path* p, denoted $\ell(p)$, is the number of arrows in p.

The distance in a directed graph can now be defined in terms of the length of a path. Let $P_{l_1}^{l_2} = P(l_1, l_2)$ be the set of all paths from l_1 to l_2 in a directed graph. Then the *distance between* l_1 *and* l_2, denoted $\rho(l_1, l_2)$, is defined as follows:

$$\rho(l_1, l_2) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } l_1 = l_2 \\ min\,\{\ell(p) : p \in P_{l_1}^{l_2}\} & \text{if } P_{l_1}^{l_2} \neq \phi \\ \infty & \text{otherwise} \end{cases}$$

[4] In a four-dimensional spacetime structure, as mentioned above, it would be a relation like *in the causal future of*.

[5] A general account on metric spaces can be found, e.g., in [78] (Chapter 7).

[6] Note that we here use the term location instead of node or vertex. We have chosen for this because, on the one hand, there exists no unique terminology in graph theory and, on the other hand, it fits better to our terminology introduced so far.

Observe that the distance between a location l_1 and a location l_2 will, in general, not be the same as the distance between l_2 and l_1, that is, the distance function for arbitrary directed graphs will not be symmetric. This is because the sets of paths need not be the same and, therefore, the minimum length can also vary. For strongly connected directed graphs, for example, the distance function will not be a metric because the property of symmetry is missing. In completely connected directed graphs the situation is quite different. The two sets of paths will be the same, i.e., $P(l_1, l_2) = P(l_2, l_1)$ because the incidence function is symmetric. Thus, the distance function will also be symmetric and moreover constitutes a metric.

Derived from the general distance function in a directed graph, the diameter of such a graph constitutes a further significant notion although it is restricted to strongly connected directed graphs only. This is because otherwise the diameter would always be infinite because at least two locations will exist for which there is no path. Hence, let G be a strongly connected directed graph. The *diameter of* G, denoted $\mho(G)$, is the largest distance between any two locations in G, i.e.,

$$\mho(G) \stackrel{\text{def}}{=} max\,\{\rho(l_1, l_2) : \text{for all } l_1, l_2 \in L\}$$

The diameter of a strongly connected directed graph will become important in part II where properties of the corresponding systems will depend on the "number of hops", i.e., the distance between locations and the diameter of the underlying graph.

3.1.2 A PRAGMATICAL PERSPECTIVE

Considering space from a pragmatical point of view it is indispensable here to include networks. Networks comprise static as well as dynamic aspects. For example, when regarding fault-tolerance within distributed real-time systems we can distinguish between the *existence* of processors and their interconnection by (physical) channels—this constitutes static properties—and the *correctness* or dually failure of processors and (physical) channels—which constitutes dynamic properties—(cf. Cristian et al. [20]).

Separating static from dynamic properties leads to a view on space and networks that is similar to our view on time and clocks. In section 2.1.2, a clock has been regarded as a source of knowledge of time and it has mathematically been defined as a time-view function. Similarly we will regard a network[7],

[7]Note that the notion of a *network* here not only comprises physical networks but it also includes logical networks.

in general, as a *source of knowledge of space*, that is, a device that can be used to refer to single points in space, e.g., for space stamping events—for instance, with the identifier of the corresponding processor—or to measure space distances between occurrences of events. Mathematically it will then be defined as a *space-view function* (see below).[8] The locative counterparts of hardware clocks will now be provided by physical networks.

NETWORKS AND SPACE

Two kinds of networks can be distinguished: A *physical network*, that is, a hardware device consisting of a number of processors (providing unique identifications and interconnected by physical channels) in order to identify the progression in *physical space* and a *logical network*, that is, a software device which is based on some mechanism in order to measure the progression in *logical space*.

Independent of being a physical or logical network it can mathematically be represented as a function from a set of locations to the direct product of a set of processors and the powerset of a set of (physical) channels. Therefore, we define such a function to be a mapping N from a so called *reference space* L_{ref} to a so called *network space* L_{net}, i.e., $N : L_{ref} \longrightarrow L_{net}$ where $L_{net} = P \times \wp(C)$ and P is a set of processors (processor identifiers) and C is a set of physical channels (see also formal definitions in chapter 5). We will call such a function *space-view function* or simply *space-view*.[9] It may be sufficient to have finite and discrete sets because physical networks are built from a finite number of processors and channels. Thereby, it is important that the cardinality of L_{ref} is at least the cardinality of P, i.e., $|P| \leq |L_{ref}|$.[10]

CORRECTNESS

The quality of the knowledge of space depends on the quality of its source, i.e., the correctness of the corresponding network. Unfortunately a physical network will not always be correct, that is, $N(l) = (l, C_l)$ not for all $l \in L_{ref}$ where $C_l \subseteq \widehat{C_l} \subseteq C$ and $\widehat{C_l}$ is the set of *all* channels associated with location l and C_l is the set of *all correct* channels associated with location l (cf. our

[8] It is important for further discussion not to confuse our usage of the term "network" with classical common usage, e.g., Sloman and Kramer [82]. Classically a network is regarded as a graph, we consider it as a function.

[9] Note the nice analogy between time-view functions as introduced in section 2.1.2 and space-view functions.

[10] We will assume coincidence between the elements in L_{ref} and in P. This is the same as with the elements of T_{ref} and T_{clock} in the temporal discourse (cf. section 2.1.2).

process view in section 1.2).[11] Moreover, due to failures of processors or physical channels physical networks may not even be connected, that is, there exists a path from l_1 to l_2 for all $l_1, l_2 \in L_{ref}$. Hence, we get quite often a picture like that in figure 3.2 where disconnected components in L_{net} can be observed.[12]

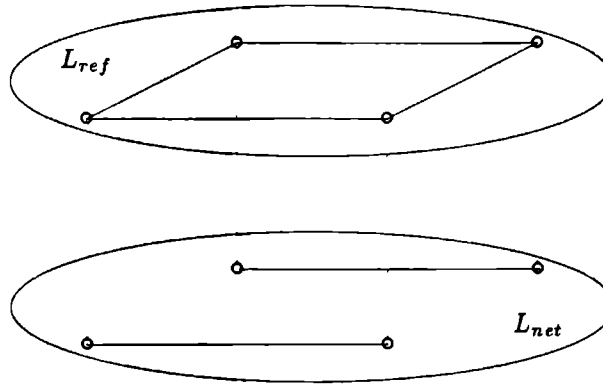


Figure 3.2: A Network Sample

A network must be presupposed to be at least connected because otherwise quite complicated situations may arise. For example, if a group of processors—perhaps only one processor—is disconnected from the rest of processors it will neither be possible to make information available to that group nor will it be possible to get information from that group.[13]

The quality of the source of knowledge of space will be considered here in a similar way as it has been done for the quality of clocks (cf. section 2.1.2), i.e., the quality of a network (space-view function) can be divided into three cases: a connected network $N : L_{ref} \longrightarrow L_{net}$ is called

- ▷ *correct at location* l iff $N(l) = (l, C_l)$ and $C_l = \widehat{C_l}$,
- ▷ *d-valent at location* l iff $N(l) = (l, C_l)$ and $C_l \subset \widehat{C_l}$ and $|C_l| = d$, and

[11] Note that correctness of a processor is modelled here by writing its identification l, say, in the pair (l, C_l). The processor identification may be thought of as contained in a cell in the main memory.

[12] Note that here and in subsequent figures we shall draw lines instead of symmetric vectors.

[13] Note that above we have implicitly assumed an arbitrary but fixed time point. However, analogous considerations will apply in the case of a dynamically evolving network, i.e., the space-view will then depend on time (cf. chapter 8).

▷ *incorrect at location* l otherwise.

Thereby, $0 < d < \left|(\widehat{C_l})\right|$ is an arbitrary but fixed number of channels indicating some redundancy at a location, i.e., the number of direct links at a location.

A connected network N is called *correct* iff N is correct at all $l \in L_{ref}$ and it is called *d-valent* iff N is d-valent at all $l \in L_{ref}$. A typical example for a connected 2-valent network is a ring network (see below).

EXAMPLE 3.1 Let be given some physical network $N : L_{ref} \longrightarrow L_{net}$ where $L_{net} = P \times \wp(C)$ and P is a set of processors and C is a set of channels. Furthermore, let l_1, l_2, l_3, and l_4 denote the four locations, respectively. Suppose that at all locations $l_1, \ldots, l_4$ the corresponding processors and channels are all correct, i.e., $N(l_1) = (l_1, \{c_1, c_4\})$, $N(l_2) = (l_2, \{c_2, c_1\})$, $N(l_3) = (l_3, \{c_3, c_2\})$, and finally $N(l_4) = (l_4, \{c_4, c_3\})$. This defines a correct network and, in particular, a ring. An illustration is given in figure 3.3.

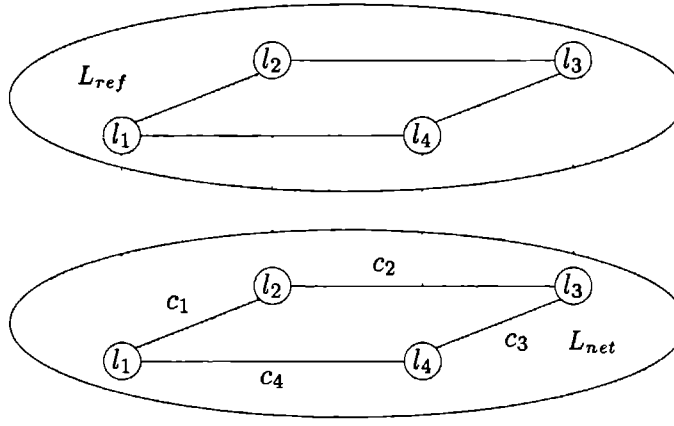


Figure 3.3: A Correct Network

Now suppose that at location l_3 all channels will have failed—for instance, at a future time point w.r.t. the correct network above thus assuming here some progression in time—, i.e., $N(l_3) = (l_3, \emptyset)$. Moreover, suppose that the space-view at location l_3 is different from the space-view at location l_2 in the sense that the common channel c_2 is considered correct at l_2 but incorrect at l_3. Similarly suppose that the space-view at l_3 is different from the one at l_4 in the sense that the common channel c_3 is considered correct at l_4 but incorrect at l_3.

Only at location l_1 it is supposed that all is correct. In terms of the space-view function the whole scenario can be described as follows: $N(l_1) = (l_1, \{c_1, c_4\})$, $N(l_2) = (l_2, \{c_2, c_1\})$, $N(l_3) = (l_3, \emptyset)$, and $N(l_4) = (l_4, \{c_4, c_3\})$.

According to our definitions above, such a network is, locally regarded, incorrect at location l_3 but correct at all other locations. From a global point of view, such a network is incorrect although a connected component consisting of locations l_1, l_2, and l_4 exists. So, take care not to confuse local and global views. An illustration is given in figure 3.4.

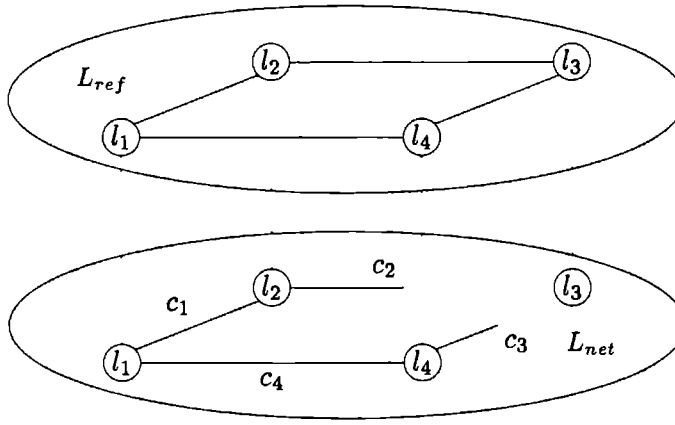


Figure 3.4: An Incorrect Network

Hence, we can conclude that not only processors and physical channels can fail but also the space-view can differ from location to location. The latter case constitutes the well-known problem of processor group membership (cf. chapter 8). Without any precautions a physical network where processors and channels are sensitive to errors may be useless for distributed real-time systems because coordinated decentralized activities "in order to cooperate to achieve an overall goal" cannot be guaranteed (cf. section 1.2). ◇

When correct physical networks cannot be guaranteed for the execution time of a distributed real-time system it will be necessary to make the system tolerant against processor and channel failures. These may happen, for example, because a channel breaks down or because the disk system associated with a processor has failed to deliver its service such that the processor stops execution (see, e.g., Coesmans and Wieczorek [16]) or because a processor itself has become faulty. To avoid such failures many different techniques are

known in the literature (see, e.g., Schepers [79] for an overview of paradigms for fault-tolerant systems). Two particular chapters in part II of this thesis will also be devoted to these problems: in chapter 7 we investigate the problem of information diffusion in a network in case of failing processors and in chapter 8 we investigate the problem of processor group membership in case of failing and restarting processors.

3.1.3 STANDARD TOPOLOGY WITH LOCATIVE DISTANCE

For the specification and verification of distributed real-time systems space as well as networks will be needed. Space itself will be used twice: as an abstract notion on a meta-level to reason about events and as a concrete notion on an object-level as observable events of space-view functions. The meta-level space notion will be our locative reference space or, as it would be called in modal logic, our locative frame (see below). The object-level space notion will come into existence by so called fault-hypotheses on the corresponding network (for more details see part II).[14]

Our locative reference space consists of a set L of locations together with a locative reachability relation.[15] As with time, we will add a *locative distance function* and a *locative metric domain.* Using first-order predicate logic with identity and a two-place predicate $R_{\mathrm{L}}^{\approx} \subseteq L \times L$ expressing locative reachability we can specify the properties as follows: (let, thereby, l, l_1, l_2, and l_3 be variables ranging over the set L of locations)

1. $\forall l : R_{\mathrm{L}}^{\approx}\, l\, l$ (Reflexivity)
2. $\forall l_1, l_2 : R_{\mathrm{L}}^{\approx}\, l_1\, l_2 \rightarrow R_{\mathrm{L}}^{\approx}\, l_2\, l_1$ (Symmetry)
3. $\forall l_1, l_2, l_3 : R_{\mathrm{L}}^{\approx}\, l_1\, l_2 \wedge R_{\mathrm{L}}^{\approx}\, l_2\, l_3 \rightarrow R_{\mathrm{L}}^{\approx}\, l_1\, l_3$ (Transitivity)

   The properties of reflexivity, symmetry, and transitivity together ensure that $R_{\mathrm{L}}^{\approx}$ is an equivalence relation on the set L of locations. Locations are equivalent in the sense that they are provided statically with the same possibilities to reach possibly other locations. The equivalence classes are then given by the strongly connected components.

   For the moment, we make the range of the locative distance function $\rho_{\mathrm{L}} : L \times L \longrightarrow \mathbb{ID}_{\mathrm{L}}$ application-dependent but require some minimal properties of the locative metric domain $\mathbb{ID}_{\mathrm{L}}$. The axiomatization of our

[14] The distinction between meta-level space notion and object-level space notion has been inspired by Dov Gabbay's metabox concept [30].

[15] We will call our locative reference space with the topological properties as given here the *standard topology of space.*

distance function and its range will, therefore, be very similar to that of the temporal distance function and the temporal metric domain (cf. chapter 2). Again we will make use of first-order predicate logic with identity and a two-place function symbol $\oplus : \mathbb{D}_L \times \mathbb{D}_L \longrightarrow \mathbb{D}_L$: (let, thereby, l_1, l_2, and l_3 be variables ranging over the set L of locations and ζ, ζ_1, ζ_2, ζ_3 be variables ranging over the set $\mathbb{D}_L$ of space distances, and let $0 \in \mathbb{D}_L$ be a constant locative distance)

4. $\forall l_1, l_2 : \rho_L(l_1, l_2) = 0 \leftrightarrow l_1 = l_2$ (Zero Distance)

5. $\forall l_1, l_2 : \rho_L(l_1, l_2) = \rho_L(l_2, l_1)$ (Symmetry)

6. $\forall l_1, l_2, l_3 : R^{\approx}_L\, l_1\ l_2 \wedge R^{\approx}_L\, l_2\ l_3 \rightarrow \rho_L(l_1, l_3) \leq \rho_L(l_1, l_2) \oplus \rho_L(l_2, l_3)$
(Conditional Inequality)

Note that the condition in the antecedent above guarantees that all distances in the succedent above, i.e., $\rho_L(l_1, l_3)$, $\rho_L(l_1, l_2)$, and $\rho_L(l_2, l_3)$ do exist.

7. $\forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = 0 \rightarrow (\zeta_1 = 0 \wedge \zeta_2 = 0)$ $(\mathbb{D}^0_L)$

8. $\forall \zeta : \zeta \oplus 0 = \zeta$ (Unit Element)

9. $\forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_2 = \zeta_1 \oplus \zeta_3 \rightarrow \zeta_2 = \zeta_3)$ (Left Injectivity)

10. $\forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_3 = \zeta_2 \oplus \zeta_3 \rightarrow \zeta_1 = \zeta_2)$ (Right Injectivity)

11. $\forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_2) \oplus \zeta_3 = \zeta_1 \oplus (\zeta_2 \oplus \zeta_3)$ (Associativity)

12. $\forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = \zeta_2 \oplus \zeta_1$ (Commutativity)

13. $\forall \zeta_1, \zeta_2 : \exists \zeta : \zeta_1 = \zeta_2 \oplus \zeta \vee \zeta_2 = \zeta_1 \oplus \zeta$ (Absolute Difference)

3.2 METRIC LOCATIVE LOGIC

Similar to time we will treat space: for the specification of qualitative and quantitative properties of distributed systems and for reasoning about them a metric locative logic will be introduced.

In locative logic (LL), qualitative locative operators such as $\Box$ (everywhere) and $\Diamond$ (somewhere) restrict reasoning over the locative universe by requiring a certain relationship between the locations at issue. If such a relationship does not exist nothing can be said about the existence of properties at those locations. For reasons of symmetry with the temporal calculus, we also introduce some qualitative extensions such as Δ (everywhere else) in locative logic.

Also in the case of locative logic the pure qualitative approach will not suffice because it will be necessary to reason about distances in a distributed system (see, for instance, chapter 7 for an application of the locative distance as a means for reasoning about the number of "hops").

Similar to the temporal case (see chapter 2) we have added indices to the locative operators to indicate the distance from the actual locative reference point. This makes it possible to specify properties relevant to distributed systems: for instance, two events can be related to one another by an a priori known space bound on the locative distance between their occurrences, e.g., events occurring at locations lying within the diameter of the network (cf. part II).

3.2.1 LANGUAGE

The language of metric locative logic (MLL) as used in this chapter is a first-order language with identity where quantification over the metric domain $\mathbb{D}_L$ will be allowed. Opposed to that quantification over the set L of locations will be disallowed. The following special symbols and sets of symbols will be assumed:

- ▷ *individual constant symbols*: an enumerable set $\mathcal{A}_C$ of constant symbols containing a special symbol 0
- ▷ *individual variable symbols*: an enumerable set $\mathcal{A}_V$ of variable symbols
- ▷ *operation symbols*: an infix function symbols $\oplus$, $=$, $<$ all of arity 2
- ▷ *propositional variable symbols*: an enumerable set $\mathcal{A}_P$ of constant predicate symbols
- ▷ *quantifiers*: $\forall$, $\exists$
- ▷ *propositional connectives*: $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
- ▷ *locative connectives*: $\boxdot^L$, $\diamondsuit^L$, $\triangle^L$, $\triangledown^L$, $\blacktriangle^L$, $\blacktriangledown^L$

  As in the temporal language, we shall indicate that a particular locative connective is reflexive, i.e., refers to the reflexive closure of the underlying accessibility relation by writing a circle within the corresponding connective.

The priorities of the propositional and locative connectives are defined as usual in the following order (within one item the priorities are the same):

1. $\neg$, $\boxdot^{\mathsf{L}}$, $\diamondsuit^{\mathsf{L}}$, $\triangle^{\mathsf{L}}$, $\triangledown^{\mathsf{L}}$, $\mathbin{\vartriangle\!\!\!\!-}^{\mathsf{L}}$, $\mathbin{\triangledown\!\!\!\!-}^{\mathsf{L}}$ (highest priority)
2. $\wedge$, $\vee$
3. $\rightarrow$, $\leftrightarrow$ (lowest priority)

The terms in metric locative logic are used to denote elements of the locative metric domain. Therefore, we will only need individual constant and variable symbols and the operation symbol:

DEFINITION 3.1 (WELL-FORMED TERMS) The set of well-formed terms in metric locative logic is the minimal set of terms closed under the following formation rules:

1. An individual constant symbol is a well-formed term.
2. An individual variable symbol is a well-formed term.
3. If η_1 and η_2 are well-formed terms then $\eta_1 \oplus \eta_2$ is a well-formed term.

$\diamondsuit$

The last rule corresponds to the above mentioned need to add two distances from the locative metric domain to get a new distance in that domain.

DEFINITION 3.2 (WELL-FORMED FORMULAE) The set of well-formed formulae in metric locative logic is the minimal set of formulae closed under the following formation rules:

1. A propositional variable symbol is a well-formed formula.
2. The propositional connective $\perp$ (false) is a well-formed formula.
3. If φ_1 and φ_2 are well-formed formulae then $\varphi_1 \rightarrow \varphi_2$ is a well-formed formula.
4. If η_1 and η_2 are well-formed terms then $\eta_1 = \eta_2$ and $\eta_1 < \eta_2$ is a well-formed formula.
5. If η is a well-formed term and φ is a well-formed formula then $\boxdot^{\mathsf{L}}_{=\eta}\, \varphi$, $\triangle^{\mathsf{L}}_{=\eta}\, \varphi$, and $\mathbin{\vartriangle\!\!\!\!-}^{\mathsf{L}}_{=\eta}\, \varphi$ are well-formed formulae.
6. If ζ is an individual variable symbol and φ is a well-formed formula then $\forall \zeta : \varphi$ is a well-formed formula.

◇

The language presented here is, in fact, a first-order extension of the metricated polymodal language $PML(R^{\mathrm{U}}_{\mathrm{L}}, R^{\approx}_{\mathrm{L}}, =_{\mathrm{L}}, \neq_{\mathrm{L}})$ with the universal relation $R^{\mathrm{U}}_{\mathrm{L}}$ on L, the locative reachability relation $R^{\approx}_{\mathrm{L}}$ on L, the relation $=_{\mathrm{L}}$ of equality on L, and the relation $\neq_{\mathrm{L}}$ of inequality on L. Informally the basic locative operators have the following meaning:

- ▷ *Everywhere in the Class (at a certain distance)* $(\boxdot^{\mathrm{L}}_{=\eta}\ \varphi)$ Formula φ is true at all locations that are reachable from the actual locative reference point and whose distance to that point is equal to the value of η.

- ▷ *Everywhere Else (at a certain distance)* $(\triangle^{\mathrm{L}}_{=\eta}\ \varphi)$ Formula φ is true at all locations that are different from the actual locative reference point and whose distance to that point is equal to the value of η. Note that this formula is equivalent to $\bot$ (false) if η evaluates to 0.

- ▷ *Everywhere (at a certain distance)* $(\boxplus^{\mathrm{L}}_{=\eta}\ \varphi)$ Formula φ is true at all locations whose distance to the actual locative reference point is equal to the value of η.

Non-primitive metric locative operators, e.g., dual versions of the above mentioned locative operators can be defined in terms of the basic ones. We will give a list of the most important metric locative operators in section 3.2.3 below.

3.2.2 FORMAL SEMANTICS

As usual we give a Kripke style semantics for our locative language, that is, we first introduce the notions of a locative frame, a metric locative frame, and a metric locative model and define afterwards the semantics of the formulae inductively on the complexity of their structure.

DEFINITION 3.3 (LOCATIVE FRAME) Let L be a non-empty set of elements (*locations*) and $R^{\approx}_{\mathrm{L}}$ be a binary relation on L (*locative reachability*). Then, the structure $(L, R^{\approx}_{\mathrm{L}})$ is called a *locative frame* or, for short, an *L-frame.* ◇

In the sequel, we will presuppose the properties of the standard topology with locative distance as presented in section 3.1.3. As with the temporal case, we shall make no distinction between the alphabet (syntactic domain) $\mathcal{A}_C$ and the semantic domain $\mathbb{D}_{\mathrm{L}}$.

DEFINITION 3.4 (METRIC LOCATIVE FRAME) Let $\mathcal{F}_{\mathrm{L}} = (L, R^{\approx}_{\mathrm{L}})$ be a locative frame and let $\mathbb{D}_{\mathrm{L}}$ be a non-empty set of elemens (*locative metric domain*). Let $\rho_{\mathrm{L}} : L \times L \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a $\mathbb{D}_{\mathrm{L}}$-valued function (*locative distance function*), $\oplus : \mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}} \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a dyadic infix operator, and $0 \in \mathbb{D}_{\mathrm{L}}$ (*unity element w.r.t.* $\oplus$). Then, the structure $(\mathcal{F}_{\mathrm{L}}, \mathbb{D}_{\mathrm{L}}, \rho_{\mathrm{L}}, \oplus, 0)$ is called a *metric locative frame* or, for short, a *metric L-frame.* ◇

DEFINITION 3.5 (EVALUATION OF TERMS) Let L be a set of locations and $l \in L$. Let Σ_{L} be the set of well-formed terms and $\alpha_{\mathrm{L}} : L \times \Sigma_{\mathrm{L}} \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a function assigning to each location and well-formed term a distance. Then, α_{L} is defined inductively on the structure of terms in the following way:

$$\alpha_{\mathrm{L}}(l,\eta) = \begin{cases} d & \text{if } \eta = d \text{ and } d \in \mathcal{A}_C \\ d & \text{if } \eta = \zeta \text{ and } \zeta \in \mathcal{A}_V \\ & \quad \text{and } \zeta \in dom(\alpha_{\mathrm{L}}(l)) \\ & \quad \text{and the value of } \zeta \text{ is } d \text{ at location } l \\ \alpha_{\mathrm{L}}(l,\eta_1) \oplus \alpha_{\mathrm{L}}(l,\eta_2) & \text{if } \eta = \eta_1 \oplus \eta_2 \text{ is a compound term} \end{cases}$$

The function α_{L} is called a *locative binding.* If α_{L} has the property that $\alpha_{\mathrm{L}}(l,\eta) = \alpha_{\mathrm{L}}(l',\eta)$ for all $l, l' \in L$ and all terms η then it is called a *rigid locative binding.* ◇

DEFINITION 3.6 (METRIC LOCATIVE MODEL) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{F}^{|\cdot|}_{\mathrm{L}} = (L, R^{\approx}_{\mathrm{L}}, \mathbb{D}_{\mathrm{L}}, \rho_{\mathrm{L}}, \oplus, 0)$ be a metric locative frame. Let α_{L} be a rigid locative binding and let $\mathcal{I}_{\mathrm{L}}$ be a function assigning to each propositional variable $p \in \mathcal{A}_P$ a subset of L on which p is true (*locative interpretation function*), i.e., $\mathcal{I}_{\mathrm{L}} : \mathcal{A}_P \longrightarrow \wp(L)$. Then, the structure $(\mathcal{F}^{|\cdot|}_{\mathrm{L}}, \alpha_{\mathrm{L}}, \mathcal{I}_{\mathrm{L}})$ is called a *metric locative model* or, for short, a *metric L-model.* ◇

DEFINITION 3.7 (LOCATIVE SATISFIABILITY) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{M}^{|\cdot|}_{\mathrm{L}} = (\mathcal{F}^{|\cdot|}_{\mathrm{L}}, \alpha_{\mathrm{L}}, \mathcal{I}_{\mathrm{L}})$ be a metric locative model and let $l \in L$. Then, a well-formed formula φ *holds in a metric locative model* $\mathcal{M}^{|\cdot|}_{\mathrm{L}}$ *at location* l or φ *is satisfied in a metric locative model* $\mathcal{M}^{|\cdot|}_{\mathrm{L}}$ *at location* l, notation $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \varphi$, is defined inductively as follows:

(1) $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \not\models_{\mathrm{LL}} \bot$

(2) $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} p$ iff $l \in \mathcal{I}_{\mathrm{L}}(p)$ for $p \in \mathcal{A}_P$

(3) $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \varphi_1 \rightarrow \varphi_2$ iff if $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \varphi_1$ then $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \varphi_2$

(4) $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \eta_1 = \eta_2$ iff $\alpha_{\mathrm{L}}(l,\eta_1) = \alpha_{\mathrm{L}}(l,\eta_2)$

(5) $\mathcal{M}^{|\cdot|}_{\mathrm{L}}, l \models_{\mathrm{LL}} \eta_1 < \eta_2$ iff $\alpha_{\mathrm{L}}(l,\eta_1) < \alpha_{\mathrm{L}}(l,\eta_2)$

(6) $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l \models_{\mathrm{LL}} \boxdot_{=\eta}^{\mathrm{L}} \varphi$ iff for all $l' \in L$:
if $R_{\mathrm{L}}^{\approx}\ l\ l'$ and $\rho_{\mathrm{L}}(l,l') = \alpha_{\mathrm{L}}(l,\eta)$
then $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l' \models_{\mathrm{LL}} \varphi$

(7) $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l \models_{\mathrm{LL}} \Delta_{=\eta}^{\mathrm{L}} \varphi$ iff $0 \neq \alpha_{\mathrm{L}}(l,\eta)$
and
for all $l' \in L$:
if $l \neq_{\mathrm{L}} l'$ and $\rho_{\mathrm{L}}(l,l') = \alpha_{\mathrm{L}}(l,\eta)$
then $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l' \models_{\mathrm{LL}} \varphi$

(8) $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l \models_{\mathrm{LL}} \triangleq_{=\eta}^{\mathrm{L}} \varphi$ iff for all $l' \in L$:
if $R_{\mathrm{L}}^{\mathrm{U}}\ l'\ l$ and $\rho_{\mathrm{L}}(l,l') = \alpha_{\mathrm{L}}(l,\eta)$
then $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l' \models_{\mathrm{LL}} \varphi$

(9) $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, l \models_{\mathrm{LL}} \forall\zeta : \varphi$ iff for all α_{L}' different from α_{L} just on ζ:
$\mathcal{M}_{\mathrm{L}}^{\mid\mid}[\alpha_{\mathrm{L}}'/\alpha_{\mathrm{L}}], l \models_{\mathrm{LL}} \varphi$

where $R_{\mathrm{L}}^{\mathrm{U}}$ is the universal relation $L \times L$. The model $\mathcal{M}_{\mathrm{L}}^{\mid\mid}[\alpha_{\mathrm{L}}'/\alpha_{\mathrm{L}}]$ results from model the $\mathcal{M}_{\mathrm{L}}^{\mid\mid}$ by replacing the binding α_{L} with the new binding α_{L}', i.e., $\mathcal{M}_{\mathrm{L}}^{\mid\mid}[\alpha_{\mathrm{L}}'/\alpha_{\mathrm{L}}] = (\mathcal{F}_{\mathrm{L}}^{\mid\mid}, \alpha_{\mathrm{L}}', \mathcal{I}_{\mathrm{L}})$.[16] ◇

DEFINITION 3.8 (LOCATIVE VALIDITY) Let $\mathcal{M}_{\mathrm{L}}^{\mid\mid} = (\mathcal{F}_{\mathrm{L}}^{\mid\mid}, \alpha_{\mathrm{L}}, \mathcal{I}_{\mathrm{L}})$ be a metric locative model.

1. A well-formed formula φ *is valid in a metric locative model* $\mathcal{M}_{\mathrm{L}}^{\mid\mid}$ or, for short, *L-valid in* $\mathcal{M}_{\mathrm{L}}^{\mid\mid}$, notation $\mathcal{M}_{\mathrm{L}}^{\mid\mid} \models_{\mathrm{LL}} \varphi$, iff for all $t \in L$: $\mathcal{M}_{\mathrm{L}}^{\mid\mid}, t \models_{\mathrm{LL}} \varphi$.

2. A well-formed formula φ *is valid*, notation $\models_{\mathrm{LL}} \varphi$, iff for all $\mathcal{M}_{\mathrm{L}}^{\mid\mid}$: $\mathcal{M}_{\mathrm{L}}^{\mid\mid} \models_{\mathrm{LL}} \varphi$.

◇

3.2.3 DEDUCTION

We will confine ourselves here to mention definitions, axioms, and rules that are basic to our metric locative logic and that extend classical propositional calculi with properties for the locative distance function and the locative metric domain (see above for their statement in first-order predicate logic). Note that we are not aiming here at a complete axiomatization of metric locative logic.

[16] Note that the universal relation $R_{\mathrm{L}}^{\mathrm{U}}$ has only been introduced for reasons of analogy between modal operators and relations on the corresponding universe.

DEFINITIONS Note that the basic operators were reflexive ones. This is different from the temporal case where the irreflexive versions have been chosen to be basic. This is because we shall need the properties of an equivalence relation in locative logic.

1. $\boxdot^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \forall \zeta : \boxdot^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere in the Class)
2. $\diamondsuit^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \exists \zeta : \diamondsuit^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere in the Class)
3. $\diamondsuit^{\mathrm{L}}_{=\eta} \varphi \stackrel{\mathrm{def}}{=} \neg \boxdot^{\mathrm{L}}_{=\eta} \neg \varphi$
4. $\triangle^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \forall \zeta : 0 < \zeta \rightarrow \triangle^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere Else)
5. $\triangledown^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \exists \zeta : 0 < \zeta \wedge \triangledown^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere Else)
6. $\triangledown^{\mathrm{L}}_{=\eta} \varphi \stackrel{\mathrm{def}}{=} \neg \triangle^{\mathrm{L}}_{=\eta} \neg \varphi$
7. $\circledcirc^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \forall \zeta : \circledcirc^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere)
8. $\circledcirc^{\mathrm{L}}_{<\eta} \varphi \stackrel{\mathrm{def}}{=} \forall \zeta : \zeta < \eta \rightarrow \circledcirc^{\mathrm{L}}_{=\zeta} \varphi$
9. $\circledtriangledown^{\mathrm{L}} \varphi \stackrel{\mathrm{def}}{=} \exists \zeta : \circledtriangledown^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere)
10. $\circledtriangledown^{\mathrm{L}}_{=\eta} \varphi \stackrel{\mathrm{def}}{=} \neg \circledcirc^{\mathrm{L}}_{=\eta} \neg \varphi$
11. $\circledtriangledown^{\mathrm{L}}_{<\eta} \varphi \stackrel{\mathrm{def}}{=} \neg \circledcirc^{\mathrm{L}}_{<\eta} \neg \varphi$
12. $\circledtriangledown^{\mathrm{L}}_{\leq\eta} \varphi \stackrel{\mathrm{def}}{=} \circledtriangledown^{\mathrm{L}}_{=\eta} \varphi \vee \circledtriangledown^{\mathrm{L}}_{<\eta} \varphi$

AXIOMS The axioms for the qualitative properties are all well-known from the literature [7, 44], the axioms for the locative metric domain are the same as in the temporal case, and the axioms for the locative distance function are slightly different from the temporal distance function in that we now have a conditional inequality instead of the conditional equality.

1. $\vdash_{\mathrm{LL}} \boxdot^{\mathrm{L}} p \rightarrow p$ (Reflexivity)
2. $\vdash_{\mathrm{LL}} p \rightarrow \boxdot^{\mathrm{L}} \diamondsuit^{\mathrm{L}} p$ (Symmetry)
3. $\vdash_{\mathrm{LL}} \boxdot^{\mathrm{L}} p \rightarrow \boxdot^{\mathrm{L}} \boxdot^{\mathrm{L}} p$ (Transitivity)
4. $\vdash_{\mathrm{LL}} \boxdot^{\mathrm{L}} (\varphi_1 \rightarrow \varphi_2) \rightarrow (\boxdot^{\mathrm{L}} \varphi_1 \rightarrow \boxdot^{\mathrm{L}} \varphi_2)$ (Distributivity)
5. $\vdash_{\mathrm{LL}} \forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = 0 \rightarrow (\zeta_1 = 0 \wedge \zeta_2 = 0)$ $(\mathbb{ID}^0_{\mathrm{L}})$
6. $\vdash_{\mathrm{LL}} \forall \zeta : \zeta \oplus 0 = \zeta$ (Unit Element)
7. $\vdash_{\mathrm{LL}} \forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_2 = \zeta_1 \oplus \zeta_3 \rightarrow \zeta_2 = \zeta_3)$ (Left Injectivity)
8. $\vdash_{\mathrm{LL}} \forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_3 = \zeta_2 \oplus \zeta_3 \rightarrow \zeta_1 = \zeta_2)$ (Right Injectivity)
9. $\vdash_{\mathrm{LL}} \forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_2) \oplus \zeta_3 = \zeta_1 \oplus (\zeta_2 \oplus \zeta_3)$ (Associativity)

10. $\vdash_{LL} \forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = \zeta_2 \oplus \zeta_1$ (Commutativity)
11. $\vdash_{LL} \forall \zeta_1, \zeta_2 : \exists \zeta : \zeta_1 = \zeta_2 \oplus \zeta \vee \zeta_2 = \zeta_1 \oplus \zeta$ (Absolute Difference)
12. $\vdash_{LL} \forall \zeta : \mathbb{V}^L \mathbb{V}^L_{=\zeta} \top$ (Surjectivity)
13. $\vdash_{LL} p \leftrightarrow \mathbb{V}^L_{=0} p$ (Zero Distance)
14. $\vdash_{LL} \forall \zeta : [\, (p \wedge \mathbb{V}^L_{=\zeta} q) \rightarrow \mathbb{V}^L_{=\zeta} (q \wedge \mathbb{V}^L_{=\zeta} p) \,]$ (Symmetry)
15. $\vdash_{LL} \forall \zeta_1, \zeta_2 : [\, \diamond^L_{=\zeta_1} \diamond^L_{=\zeta_2} p \rightarrow \mathbb{V}^L_{\leq \zeta_1 \oplus \zeta_2} p \,]$ (Conditional Inequality)

RULES The rules are given as usual in classical modal logics, i.e., modus ponens and necessitation.

1. whenever $\vdash_{LL} \varphi_1 \rightarrow \varphi_2$ and $\vdash_{LL} \varphi_1$ then $\vdash_{LL} \varphi_2$ (Modus Ponens)
2. whenever $\vdash_{LL} \varphi$ then $\vdash_{LL} \boxdot^L_{=\tau} \varphi$ (Necessitation)

Some properties of the metric function and the metric domain that we have mentioned for the temporal calculus are also derivable in the locative calculus. But because we are interested in the combination of the locative and temporal calculi we will avoid to mention any properties here. For the combined calculus, we refer to chapter 4.

3.3 A RUNNING EXAMPLE (CONTD.)

Recall that we have started a discussion about the well-known paradigm of dining philosophers at least for two reasons: firstly, we want to *illustrate* the principal distinctions between the various logics as introduced in this first part and, secondly, we want to *demonstrate* the adequacy of our specification method. Note that comparison and conclusions will be postponed till chapter 4 when all kinds of logics will have been defined.

Upto here we have provided a specification of the dining philosophers in pure first-order predicate logic (cf. chapter 1) and in temporal logic (cf. chapter 2). We are now going to present a specification of those properties by making use of locative logic as defined above.

3.3.1 THE MODEL

An axiomatization of our model of the dining philosophers community will again comprise temporal properties as well as locative properties. This time the properties of the locative reference space have already been provided with the locative logic above. What remains to be done in this section is a characterization of the temporal structure.

TEMPORAL STRUCTURE We assume an infinite, discrete set T of time points together with a binary relation $<$ denoting *temporal precedence* between two time points, a binary relation $\neq_T$ denoting *inequality* between two time points, and a function $|\cdot|_T$ denoting the distance between two time points.

The properties of the precedence relation are given as follows: (let, thereby, t, t', t'', and t''' be variables ranging over T)

1. $\forall t : [\ \neg t < t\ ]$ (Irreflexivity)
2. $\forall t, t' : [\ t < t' \rightarrow \neg t < t'\ ]$ (Asymmetry)
3. $\forall t, t', t'' : [\ t < t' \wedge t' < t'' \rightarrow t < t''\ ]$ (Transitivity)
4. $\forall t, t' : [\ t \neq t' \rightarrow t < t' \vee t' < t\ ]$ (Connectedness)
5. $\forall t, t' : [\ t < t' \rightarrow \exists t'' : [\ t < t'' \wedge \neg\exists t''' : t < t''' < t''\ ]\ ]$
   $\wedge$ (Discreteness)
   $\forall t, t' : [\ t < t' \rightarrow \exists t'' : [\ t'' < t' \wedge \neg\exists t''' : t'' < t''' < t'\ ]\ ]$
6. $\forall t : [\ \exists t' : t < t'\ ]$ (No End)

3.3.2 THE SPECIFICATION

For the specification of the requirements of the dining philosophers community we will need additional predicates:[17] (let, thereby, t, t', and t'' be variables ranging over T)

- $eaten(t)$: has just eaten at time t
- $lh(t)$: possesses a fork in his left hand at time t
- $rh(t)$: possesses a fork in his right hand at time t

Furthermore, let us introduce for notational convenience the abbreviation $bh(t)$ for the possession of a fork in both hands at time t:

$$bh(t) \stackrel{\text{def}}{=} lh(t) \wedge rh(t)$$

Finally, the community of dining philosphers has to satisfy the following requirements:[18]

[17] Note that the predicates will be indefinite w.r.t. space (cf. section 3.1).

[18] Note again that being a member of a particular community of dining philosophers will be modelled here by the locative reachability relation. As a consequence this means that *all* dining philosophers communities have to satisfy the corresponding properties or, in other words, to recognize a set of philosophers as a community of dining philosophers implies that all the properties must be valid for them.

EAT GUARANTEE All philosophers in the community will have eaten infinitely many times:

$$\boxdot^{\mathrm{L}}[\ \forall t : \exists t' : t < t' \wedge \mathit{eaten}(t')\ ]$$

HAVING EATEN NEEDS FORKS A philosopher in the community can have eaten only if he possessed a fork in his left hand and a fork in his right hand at two successive points in time just before having eaten:

$$\begin{aligned}\boxdot^{\mathrm{L}}[\ \forall t : \mathit{eaten}(t) \ \rightarrow\ & [\ \exists t' : t' < t \wedge |(t,t')|_{\mathrm{T}} = 1 \wedge \mathit{bh}(t')\ ] \\ & \wedge \\ & [\ \exists t'' : t'' < t \wedge |(t,t'')|_{\mathrm{T}} = 2 \wedge \mathit{bh}(t'')\ ]\ ]\end{aligned}$$

EXCLUSION OF FORK POSSESSION If some philosopher possesses two forks then his neighbours may not possess two forks at the same point in time:

$$\boxdot^{\mathrm{L}}[\ \forall t : [\mathit{bh}(t) \rightarrow \neg \diamondsuit^{\mathrm{L}}_{=1}\ \mathit{bh}(t)]\ ]$$

Note that we have again (cf. chapter 1) avoided to mention any start condition for the dining philosophers community.

Chapter 4

Locative Temporal Logic

Outline

The idea of having more than one dimension for reasoning about systems has been pronounced by various people with different intentions at different times. In all but one case, to our knowledge, these dimensions have been chosen to be equal, for example, two time dimensions. Our motivation for a locative temporal logic had been given by two aspects that are common to all distributed real-time systems: distribution of computing power and actions in real-time. Events in a distributed real-time system become dependent on the location and the time point of their occurrence. Thus a space-time model seems adequate for such systems.

This chapter contains the following sections: Section 4.1 discusses our locative temporal reference space. In section 4.2, we will formally define a basic version of locative temporal logic with the additional features of locative and temporal distances. Section 4.3 provides a locative temporal logic specification of the dining philosophers paradigm as introduced in the introduction. In section 4.4, we will summarize the results from part I and draw some conclusions emerging from the several specifications of the dining philosophers paradigm.

4.1 THE LOCATIVE TEMPORAL REFERENCE SPACE

A proposition denoting a property of a distributed real-time system would be meaningless as long as we do not know both *when* to look and *where* to look.[1] This exactly provides the reason to introduce a direct product of a temporal and locative reference space (see chapter 2 on temporal logic and chapter 3 on locative logic for the properties of the corresponding spaces alone).

Although we are not interested in relativistic effects and although *space* and *time* are both *acausal*[2] we have decided to construct our reference space of locative temporal logic (LTL) from space and time. Having a locative temporal reference space at our disposal truth and falsity of propositions such as "message m has been received" will be meaningful in both respects locatively *and* temporally. Without it such propositions would be meaningless. Observe that the truth of such a proposition depends on the chosen reference point. A proposition whose truth or falsity is space- and time-dependent will be called *indefinite w.r.t. space and time* or *locatively and temporally indefinite.*[3]

Similarly a proposition such as "message m has been received at time t at location l" will be meaningful when interpreted over a locative temporal reference space where $\langle l, t \rangle$ is an element of that space. Although in this case a locative temporal reference point is meaningless because the proposition itself contains an absolute space and time reference. A proposition whose truth or falsity is space- and time-independent will be called *definite w.r.t. space and time* or *locatively and temporally definite.*[4]

Our locative temporal approach gives rise to two additional classes of propositions: a proposition such as "message m has been received at time t" will be meaningful when interpreted over a locative temporal reference space where the temporal component of a locative temporal reference point $\langle l, t' \rangle$ is now meaningless because the proposition itself contains an absolute time reference. A proposition whose truth or falsity is space-dependent and time-independent will be called *indefinite w.r.t. space and definite w.r.t. time* or

[1] This has also been observed by Le Lann in his early paper on "Distributed Systems—Towards A Formal Approach" [58].

[2] "The difference between one event and another does not depend on the mere difference of the times or the places at which they occur, but only on the differences in the nature, configuration, or motion of the bodies concerned." [65]

[3] To avoid confusion with physical or philosophical spacetime (cf. previous chapter) we prefer to say "space and time" when talking conceptually about our construction of space-time, i.e., locative and temporal reference space. In a few cases, we shall also write "space-time" but with a dash.

[4] The phrases "definite proposition" and "indefinite proposition" have been borrowed from Rescher and Urquhart [75].

locatively indefinite and temporally definite.

And finally, a proposition such as "message m has been received at location l" will be meaningful when interpreted over a locative temporal reference space where the locative component of a locative temporal reference point $\langle l', t \rangle$ is now meaningless because the proposition itself contains an absolute space reference. A proposition whose truth or falsity is space-independent and time-dependent will be called *definite w.r.t. space and indefinite w.r.t. time* or *locatively definite and temporally indefinite.*

4.1.1 SPACE AND TIME IN PERSPECTIVE

Our construction of a locative temporal reference space is the direct product of the locative and temporal reference spaces as discussed in the previous two chapters (cf. also Koole [50]). The qualitative and quantitative properties of our locative temporal reference space are directly induced by the properties of their locative and temporal correspondents, respectively. Inspired by the four types of propositions as mentioned above we will now briefly discuss several kinds of properties which we find relevant for distributed real-time systems.[5]

NO EVOLUTION

The simplest case of properties in a distributed real-time system can be identified with *one* position in space and time, i.e., the locative temporal reference point. Such properties can be described by LTL formulae that contain no locative temporal operators but perhaps some propositional connectives. A typical situation of such properties is illustrated in figure 4.1.[6]

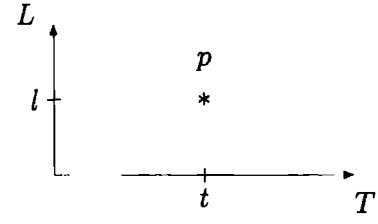


Figure 4.1: A Local Static Property

[5]For questions about the formal semantics of the formulae used in this section we refer to section 4.2.

[6]Note that in all the figures 4.1 to 4.5 L is the locative reference space, T is the temporal reference space, and p is some locatively and temporally indefinite constant predicate.

Truth or falsity of such a formula only depends on the chosen reference point, i.e., $\langle l, t \rangle$ in figure 4.1. No other positions in $L \times T$ are involved during the evaluation of such a formula.

We will call properties whose truth is fixed at the chosen locative temporal reference point *local static* because such properties are neither evolving in space nor in time. Usually such properties will be used to fix the locative temporal reference point itself, e.g., $\langle l, t \rangle$ as in figure 4.1. For example, a characterization of communication behaviour can be done by fixing the time point and the location at which a message has been sent and deriving properties about the reception of that message. Dually we can fix the time point and location at which a message has been received and derive properties about the sending of that message. If both views, i.e., the sender's view and the receiver's view coincide in the number and order of messages, presupposing their uniqueness, we can say that communication is correct (for more details see part II in this thesis).

EVOLUTION IN TIME

The next case of properties in a distributed real-time system can be identified with *two* positions in space and time: the first position being the locative temporal reference point and the second position being different from the first only in its temporal component. Such properties can be described by LTL formulae that contain locative temporal operators that affect only time. A typical situation of such properties is illustrated in figure 4.2.

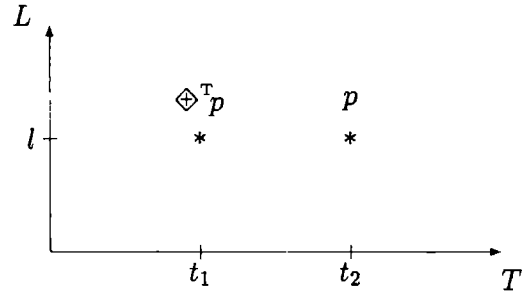


Figure 4.2: A Local Dynamic Property

Truth or falsity of such a formula depends on the chosen reference point, i.e., $\langle l, t_1 \rangle$ in figure 4.2 and on one further position in $L \times T$ which is possibly different from the chosen reference point only in its temporal component, i.e.,

$\langle l, t_2 \rangle$ in figure 4.2. No positions in $L \times T$ are involved during the evaluation of such a formula where the locative component is different from l.

We will call properties whose truth is depending only on time *local dynamic* because such properties are not evolving in space but in time. Usually such properties will be used to denote a special class of liveness properties. For example, sender and receiver in CSP [42] must synchronize their activities before being able to exchange information with their communication partner. So there might be some time difference at the sender's site between being synchronized and having succesfully terminated the exchange of information. The same will be true for the receiver's site (for more details see chapter 6 in this thesis).

EVOLUTION IN SPACE

The third case of properties in a distributed real-time system can be identified with *two* positions in space and time: the first position being the locative temporal reference point and the second position being different from the first only in its locative component. Such properties can be described by LTL formulae that contain locative temporal operators that affect only space. A typical situation of such properties is illustrated in figure 4.3.

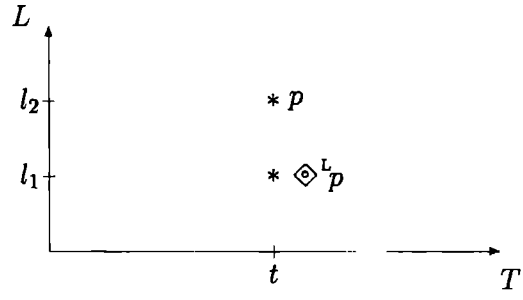


Figure 4.3: A Distributed Static Property

Truth or falsity of such a formula depends on the chosen reference point, i.e., $\langle l_1, t \rangle$ in figure 4.3 and on one further position in $L \times T$ which is possibly different from the chosen reference point only in its locative component, i.e., $\langle l_2, t \rangle$ in figure 4.3. No positions in $L \times T$ are involved during the evaluation of such a formula where the temporal component is different from t.

We will call properties whose truth is depending only on space *distributed static* because such properties are not evolving in time but in space. Such

properties can be used to denote a special class of liveness properties but liveness now understood in space. For example, if the processor associated with the locative temporal reference point is correct then one might require that if there is some other location where the corresponding processor is correct there exists some path between the two processors. In other words, the underlying surviving network must always be connected (see, for example, the fault-hypothesis on network partitioning in chapters 7 and 8).

EVOLUTION IN SPACE AND TIME

The last case of properties in a distributed real-time system can be identified with *three* positions in space and time where two different cases can be distinguished depending on the order of locative temporal operators, that is, either first evolution in space and then in time or first evolution in time and then in space. Typical situations of both cases are illustrated in figure 4.4.

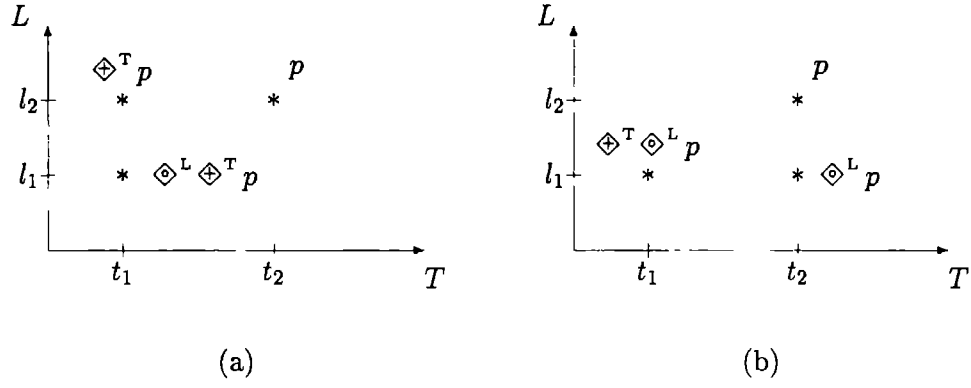


Figure 4.4: A Distributed Dynamic Property

In picture (a) above, the first position is the locative temporal reference point, the second position being different from the first in its locative component, and the third position being different from the second in its temporal component and, thus, being different from the first position in its locative and temporal components. Such properties can be described by LTL formulae that contain a pair of locative temporal operators that affect firstly space, i.e., $\diamondsuit^L$ and secondly time, i.e., $\diamondsuit^T$. Truth or falsity of such a formula depends on the chosen reference point, i.e., $\langle l_1, t_1 \rangle$ in figure 4.4, on one intermediate second position in $L \times T$, i.e., $\langle l_2, t_1 \rangle$ in figure 4.4, and on one third position in $L \times T$ which is possibly different from the chosen reference point in its locative

and temporal components, i.e., $\langle l_2, t_2 \rangle$ in figure 4.4.

In picture (b) above, the first position is again the locative temporal reference point, the second position being different from the first in its temporal component, and the third position being different from the second in its locative component and, thus, being different from the first position in its locative and temporal components. Such properties can be described by LTL formulae that contain a pair of locative temporal operators that affect firstly time, i.e., $\diamondsuit^T$ and secondly space, i.e., $\diamondsuit^L$. Truth or falsity of such a formula depends on the chosen reference point, i.e., $\langle l_1, t_1 \rangle$ in figure 4.4, on one intermediate second position in $L \times T$, i.e., $\langle l_1, t_2 \rangle$ in figure 4.4, and on one third position in $L \times T$ which is possibly different from the chosen reference point in its locative and temporal components, i.e., $\langle l_2, t_2 \rangle$ in figure 4.4.

We will call properties whose truth is depending on space as well as time *distributed dynamic* because such properties are evolving in space as well as in time. Such properties can be used to denote a special class of liveness properties but liveness now understood in space and time, e.g., if at $\langle l_1, t_1 \rangle$ predicate p holds then at $\langle l_2, t_2 \rangle$ predicate p will also hold (cf. figure 4.4). An example of this kind of properties will be the information diffusion principle as discussed in chapter 6.[7]

EVOLUTION IN SPACE AND TIME USING DISTANCES

A special class of distributed dynamic properties contains those that are not only related in a qualitative way but also by some distance. These properties can be identified with *three* positions in space and time using distances:[8] the first position being the locative temporal reference point, the second position being different from the first only in its locative component but this time at a fixed locative distance from the reference point, and the third position being different from the second only in its temporal component but this time at a fixed temporal distance. Hence, the third position is different from the locative temporal reference point in its locative and temporal component and furthermore it is at a fixed distance in space and time from the reference point. A typical situation is illustrated in figure 4.5.

Truth or falsity of such a formula depends on the chosen reference point, i.e., $\langle l_1, t_1 \rangle$ in figure 4.5, on one intermediate second position in $L \times T$, i.e.,

[7] Note that in this thesis the two cases as illustrated in figure 4.4 will be equivalent (see confluency property in section 4.2.3 on deduction).

[8] According to the confluency property (cf. section 4.2.3), we will discuss here only the case of first evolution in space and then in time and neglect the case of first evolution in time and then in space.

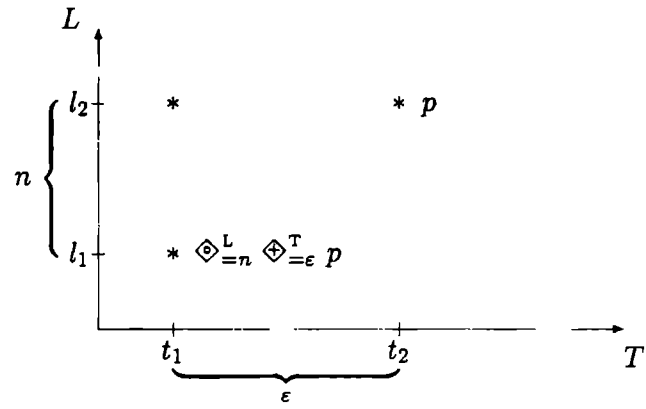


Figure 4.5: A Metric Distributed Dynamic Property

$\langle l_2, t_1 \rangle$ in figure 4.4 where the distance between l_1 and l_2 is given by n, and on one third position in $L \times T$ which is possibly different from the chosen reference point in its locative and temporal components, i.e., $\langle l_2, t_2 \rangle$ in figure 4.4 where the distance between t_1 and t_2 is given by ε.

We will call properties whose truth is depending on space and time in a qualitative but also in a quantitative sense *metric distributed dynamic*. Such properties can be used to denote a special class of liveness properties, i.e., liveness properties in space and time where a locative temporal distance is indispensable, e.g., if at $\langle l_1, t_1 \rangle$ predicate p holds then at $\langle l_2, t_2 \rangle$ predicate p will also hold where the locative distance between the two positions is equal to n and the temporal distance is equal to ε (cf. figure 4.5). A typical example of metric distributed dynamic properties has been constituted by the principle of information diffusion as discussed by Cristian et al. in [20]. We shall come back to this principle in chapters 6 and 7.

Our discussion about locative and temporal distances in the context of distributed dynamic properties will, of course, analogously apply to local dynamic and distributed static properties. In case of local static properties only a distance equal to zero makes sense. But a formula containing a reflexive locative temporal operator with zero distance would be equivalent to the operand of such an operator.

4.1.2 STANDARD TOPOLOGY WITH DISTANCES

As we have seen in the two preceding chapters space as well time but also space-view as well as time-view functions (clocks) will be needed for the specification

and verification of distributed real-time systems. Space and time will be used twice: as abstract notions on a meta-level to reason about events and as concrete notions on an object-level as observable events of space-view and time-view functions, respectively. The meta-level space and time notions will constitute our locative temporal reference space or, as it would be called in modal logic, our locative temporal frame (see below). The object-level space and time notions will come into existence by so called fault-hypotheses on the corresponding clock system and communication network (for more details see part II).[9]

Our locative temporal reference space consists of the direct product of a set L of locations and a set T of time points together with two accessibility relations: one for locative accessibility and one for temporal accessibility.[10] As with space and time alone distance functions together with their metric domains will be needed. The properties of the corresponding distance function and metric domains can directly be taken from the previous two chapters. Therefore, we will avoid to repeat them here. Using first-order predicate logic with identity, a two-place predicate $R_{\mathrm{L}}^{\approx} \subseteq (L \times T) \times (L \times T)$ expressing locative reachability, and a two-place predicate $R_{\mathrm{T}}^{<} \subseteq (L \times T) \times (L \times T)$ expressing temporal precedence we can specify the properties of our locative temporal reference space:

LOCATIVE PROPERTIES Let l, l_1, l_2, l_3, and l_4 be variables ranging over the set L of locations and let $t \in T$ be an arbitrary but fixed time point:

1. $\forall l : R_{\mathrm{L}}^{\approx} \langle l, t \rangle \langle l, t \rangle$ (L-Reflexivity)
2. $\forall l_1, l_2 : R_{\mathrm{L}}^{\approx} \langle l_1, t \rangle \langle l_2, t \rangle \rightarrow R_{\mathrm{L}}^{\approx} \langle l_2, t \rangle \langle l_1, t \rangle$ (L-Symmetry)
3. $\forall l_1, l_2, l_3 : R_{\mathrm{L}}^{\approx} \langle l_1, t \rangle \langle l_2, t \rangle \wedge R_{\mathrm{L}}^{\approx} \langle l_2, t \rangle \langle l_3, t \rangle \rightarrow R_{\mathrm{L}}^{\approx} \langle l_1, t \rangle \langle l_3, t \rangle$ (L-Transitivity)

TEMPORAL PROPERTIES Let t, t_1, t_2, and t_3 be variables ranging over the set T of time points and let $l \in L$ be an arbitrary but fixed location:

1. $\forall t : \neg R_{\mathrm{T}}^{<} \langle l, t \rangle \langle l, t \rangle$ (T-Irreflexivity)
2. $\forall t_1, t_2, t_3 : R_{\mathrm{T}}^{<} \langle l, t_1 \rangle \langle l, t_2 \rangle \wedge R_{\mathrm{T}}^{<} \langle l, t_2 \rangle \langle l, t_3 \rangle \rightarrow R_{\mathrm{T}}^{<} \langle l, t_1 \rangle \langle l, t_3 \rangle$ (T-Transitivity)

[9] The distinction between meta-level space-time notion and object-level space-time notion has been inspired by Dov Gabbay's metabox concept [30].

[10] Although the term "standard topology" is perhaps better applied to four-dimensional spacetime (cf. chapter 3) we will call our locative temporal reference space with the topological properties as given in this section the *standard topology of space and time.*

3. $\forall t_1, t_2 : \langle l, t_1\rangle \neq_{\mathrm{T}} \langle l, t_2\rangle \rightarrow R^{<}_{\mathrm{T}} \langle l, t_1\rangle \langle l, t_2\rangle \vee R^{<}_{\mathrm{T}} \langle l, t_2\rangle \langle l, t_1\rangle$
(T-Connectedness)

4. $\forall t_1, t_2 : \exists t_3 : \langle l, t_1\rangle \neq_{\mathrm{T}} \langle l, t_2\rangle \wedge R^{<}_{\mathrm{T}} \langle l, t_1\rangle \langle l, t_2\rangle \rightarrow$
$\langle l, t_1\rangle \neq_{\mathrm{T}} \langle l, t_3\rangle \wedge \langle l, t_2\rangle \neq_{\mathrm{T}} \langle l, t_3\rangle \wedge$
$R^{<}_{\mathrm{T}} \langle l, t_1\rangle \langle l, t_3\rangle \wedge R^{<}_{\mathrm{T}} \langle l, t_3\rangle \langle l, t_2\rangle$ (T-Denseness)

5. $\forall t_1 : \exists t_2 : R^{<}_{\mathrm{T}} \langle l, t_1\rangle \langle l, t_2\rangle$ (No T-End)

LOCATIVE TEMPORAL PROPERTIES Let l_1, l_2, l_3, and l_4 be variables ranging over the set L of locations and let t_1, t_2, t_3, and t_4 be variables ranging over the set T of time points:[11]

1. $\forall l_1, l_2 : \forall t_1, t_2 :$
$R^{<}_{\mathrm{T}} \langle l_1, t_1\rangle \langle l_1, t_2\rangle \wedge R^{\approx}_{\mathrm{L}} \langle l_1, t_2\rangle \langle l_2, t_2\rangle \rightarrow$
$\exists l_3 : \exists t_3 : R^{\approx}_{\mathrm{L}} \langle l_1, t_1\rangle \langle l_1, t_3\rangle \wedge R^{<}_{\mathrm{T}} \langle l_1, t_3\rangle \langle l_3, t_3\rangle$
(TL-Commutativity)

   This formula expresses the fact that if there is evolution at first in the temporal sort and then in the locative sort then there exists a point such that evolution from the original point can first proceed in the locative sort and then in the temporal sort.

2. $\forall l_1, l_2 : \forall t_1, t_2 :$
$R^{\approx}_{\mathrm{L}} \langle l_1, t_1\rangle \langle l_1, t_2\rangle \wedge R^{<}_{\mathrm{T}} \langle l_1, t_2\rangle \langle l_2, t_2\rangle \rightarrow$
$\exists l_3 : \exists t_3 : R^{<}_{\mathrm{T}} \langle l_1, t_1\rangle \langle l_1, t_3\rangle \wedge R^{\approx}_{\mathrm{L}} \langle l_1, t_3\rangle \langle l_3, t_3\rangle$
(LT-Commutativity)

   This formula expresses the fact that if there is evolution at first in the locative sort and then in the temporal sort then there exists a point such that evolution from the original point can first proceed in the temporal sort and then in the locative sort.

3. $\forall l_1, l_2 : \forall t_1, t_2 :$
$R^{<}_{\mathrm{T}} \langle l_1, t_1\rangle \langle l_1, t_2\rangle \wedge R^{\approx}_{\mathrm{L}} \langle l_1, t_1\rangle \langle l_2, t_1\rangle \rightarrow$
$R^{\approx}_{\mathrm{L}} \langle l_1, t_2\rangle \langle l_2, t_2\rangle \wedge R^{<}_{\mathrm{T}} \langle l_2, t_1\rangle \langle l_2, t_2\rangle$
(LT-Confluency)

   This formula expresses the fact that if there is evolution in the temporal sort and if there is also evolution in the locative sort then there exists a point such that from the former two points there is evolution in the locative sort and in the temporal sort, respectively.

[11] The first-order conditions for those modal formulae that contain locative as well as temporal evolution have been developed in collaboration with Wiebe van der Hoek.

4.2 METRIC LOCATIVE TEMPORAL LOGIC

For the specification of qualitative and quantitative properties of distributed real-time systems and for reasoning about them metric locative temporal logic (MLTL) will be defined in this section. Locative temporal logic (LTL) is a *two-sorted* modal logic (cf. Koole [50]). It is two-sorted in the sense that two *different* universes (see above) are used to build our reference space, i.e., a two-sorted Kripke frame.

To our knowledge, the only publication considering a general framework of many-sorted modal logic can be found in [83] by Stuhlmann-Laeisz. Unfortunately, Stuhlmann-Laeisz himself has called his logic many-dimensional although he allows explicitly for a compound universe of n *different* dimensions. Also in the literature, we can find modal logics that make use of a two-dimensional universe with *equal* dimensions. Such modal logics will be called *two-dimensional*. For example, two-dimensional temporal logics as in Segerberg [80], Gabbay [29], and Fariñas [26]. A systematic investigation of the underlying general framework of many-dimensional modal logics has recently been published by Venema [87].

In locative temporal logic, we will make use of the modal operators $\Box$ (always/everywhere) and $\Diamond$ (eventually/somewhere) as introduced in chapter 2 and 3. But now these operators will restrict our reasoning over a two-sorted universe, i.e., the locative temporal universe by requiring a certain relationship between pairs of location and time point at issue. But qualitative locative temporal operators alone will not suffice when quantitative properties come into existence.

In metric locative temporal logic, restricting reasoning over the locative temporal universe has been resumed: indices have been added to the locative temporal operators to indicate the distance in space and time w.r.t. the actual locative temporal reference point. This makes it possible to specify properties relevant to distributed real-time systems: for instance, two events can be related to one another by an a priori known bound for the distance in space and time between their occurrences (see also figure 4.5 above).

4.2.1 LANGUAGE

The language of metric locative temporal logic as used in this chapter is a first-order language with identity where quantification over the metric domains $\mathbb{D}_L$ and $\mathbb{D}_T$, respectively, will be allowed. Opposed to that quantification over the set $L \times T$ of pairs of location and time point will not be allowed. Quantification will also be disallowed over the sets L and T alone. The following special

symbols and sets of symbols will be assumed:

- ▷ *individual constant space symbols*: an enumerable set $\mathcal{A}_{C_L}$ of constant space symbols containing a special symbol 0
- ▷ *individual constant time symbols*:[12] an enumerable set $\mathcal{A}_{C_T}$ of constant time symbols containing a special symbol 0
- ▷ *individual variable space symbols*: an enumerable set $\mathcal{A}_{V_L}$ of variable space symbols
- ▷ *individual variable time symbols*: an enumerable set $\mathcal{A}_{V_T}$ of variable time symbols
- ▷ *operation symbols*:[13] infix function symbols $\oplus$, $\otimes$, $=$, $<$ all of arity 2
- ▷ *propositional variable symbols*: an enumerable set $\mathcal{A}_P$ of constant predicate symbols
- ▷ *quantifiers*: $\forall$, $\exists$
- ▷ *propositional connectives*: $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
- ▷ *locative temporal connectives*: $\boxdot^{L}$, $\diamondsuit^{L}$, $\triangle^{L}$, $\triangledown^{L}$, $\circledcirc^{L}$, $\circledtriangledown^{L}$, $\boxminus^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\diamondsuit^{T}$, $\boxdot^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\diamondsuit^{T}$, $\triangle^{T}$, $\triangledown^{T}$, $\circledcirc^{T}$, $\circledtriangledown^{T}$,

  Again, we shall indicate that a particular locative temporal connective is reflexive, i.e., refers to the reflexive closure of the underlying accessibility relation by writing a circle within the corresponding connective. A minus or plus sign within a connective indicates that it is a past or future connective, respectively.

The priorities of the propositional and locative temporal connectives are defined as usual in the following order (within one item the priorities are the same):

1. $\neg$,
   $\boxdot^{L}$, $\diamondsuit^{L}$, $\triangle^{L}$, $\triangledown^{L}$, $\circledcirc^{L}$, $\circledtriangledown^{L}$,
   $\boxminus^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\diamondsuit^{T}$, $\boxdot^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\diamondsuit^{T}$, $\triangle^{T}$, $\triangledown^{T}$, $\circledcirc^{T}$, $\circledtriangledown^{T}$ (highest priority)

[12] For notational convenience, we will make no syntactical distinction between 0 as a symbol in $\mathcal{A}_{C_L}$ or $\mathcal{A}_{C_T}$. Assume that this symbol is a polymorphic, i.e., overloaded constant operator symbol.

[13] For notational convenience, we will also make no syntactical distinction between $\oplus$, $\otimes$, $=$, $<$ as symbols used in locative and temporal terms (see below), respectively. Assume that these symbols are polymorphic, i.e., overloaded dyadic operator symbols.

2. $\wedge$, $\vee$

3. $\rightarrow$, $\leftrightarrow$ (lowest priority)

The terms in metric locative temporal logic are used to denote elements of the locative and temporal metric domain, respectively. Therefore, we will distinguish between two kinds of terms:

DEFINITION 4.1 (WELL-FORMED LOCATIVE TERMS) In metric locative temporal logic, the set of well-formed locative terms is the minimal set of terms closed under the following formation rules:

1. An individual constant space symbol is a well-formed locative term.
2. An individual variable space symbol is a well-formed locative term.
3. If η_1 and η_2 are well-formed locative terms then $\eta_1 \oplus \eta_2$ is a well-formed locative term.

$\Diamond$

DEFINITION 4.2 (WELL-FORMED TEMPORAL TERMS) In metric locative temporal logic, the set of well-formed temporal terms is the minimal set of terms closed under the following formation rules:

1. An individual constant time symbol is a well-formed temporal term.
2. An individual variable time symbol is a well-formed temporal term.
3. If τ_1 and τ_2 are well-formed temporal terms then $\tau_1 \oplus \tau_2$ is a well-formed temporal term.

$\Diamond$

DEFINITION 4.3 (WELL-FORMED FORMULAE) The set of well-formed formulae in metric locative temporal logic is the minimal set of formulae closed under the following formation rules:

1. A propositional variable symbol is a well-formed formula.
2. The propositional connective $\perp$ (false) is a well-formed formula.
3. If φ_1 and φ_2 are well-formed formulae then $\varphi_1 \rightarrow \varphi_2$ is a well-formed formula.

4. If η_1 and η_2 are well-formed locative terms then $\eta_1 = \eta_2$ and $\eta_1 < \eta_2$ are well-formed formulae.

5. If τ_1 and τ_2 are well-formed temporal terms then $\tau_1 = \tau_2$ and $\tau_1 < \tau_2$ are well-formed formulae.

6. If η is a well-formed locative term and φ is a well-formed formula then $\boxdot^{\mathrm{L}}_{=\eta}\,\varphi$, $\triangle^{\mathrm{L}}_{=\eta}\,\varphi$, and $\circledcirc^{\mathrm{L}}_{=\eta}\,\varphi$ are well-formed formulae.

7. If τ is a well-formed temporal term and φ is a well-formed formula then $\boxminus^{\mathrm{T}}_{=\tau}\,\varphi$, $\boxplus^{\mathrm{T}}_{=\tau}\,\varphi$, $\triangle^{\mathrm{T}}_{=\tau}\,\varphi$, and $\circledcirc^{\mathrm{T}}_{=\tau}\,\varphi$ are well-formed formulae.

8. If ζ is an individual variable space symbol and φ is a well-formed formula then $\forall\zeta : \varphi$ is a well-formed formula.

9. If δ is an individual variable time symbol and φ is a well-formed formula then $\forall\delta : \varphi$ is a well-formed formula.

$\Diamond$

The language presented here is, in fact, a first-order extension of the metricated polymodal language $PML(R^{\mathrm{U}}_{\mathrm{L}}, R^{\mathrm{U}}_{\mathrm{T}}, R^{\approx}_{\mathrm{L}}, R^{<}_{\mathrm{T}}, =_{\mathrm{L}}, =_{\mathrm{T}}, \neq_{\mathrm{L}}, \neq_{\mathrm{T}})$ with the universal locative relation $R^{\mathrm{U}}_{\mathrm{L}}$ on $L \times T$, the universal temporal relation $R^{\mathrm{U}}_{\mathrm{T}}$ on $L \times T$, the locative accessibility relation $R^{\approx}_{\mathrm{L}}$ on $L \times T$, the temporal accessibility relation $R^{<}_{\mathrm{T}}$ on $L \times T$, the relation $=_{\mathrm{L}}$ of locative equality on $L \times T$, the relation $=_{\mathrm{T}}$ of temporal equality on $L \times T$, the relation $\neq_{\mathrm{L}}$ of locative inequality on $L \times T$, and the relation $\neq_{\mathrm{T}}$ of temporal inequality on $L \times T$.[14] Informally the basic locative temporal operators have the following meaning:

- ▷ *Everywhere in the Class (at a distance)* ($\boxdot^{\mathrm{L}}_{=\eta}\,\varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where t is an arbitrary but fixed time point and l is a location that is reachable from the actual locative reference point. The distance between the actual locative reference point and l must be equal to the value of η.

- ▷ *Everywhere Else (at a distance)* ($\triangle^{\mathrm{L}}_{=\eta}\,\varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where t is an arbitrary but fixed time point and l is a location that is different from the actual locative reference point. The distance between the actual locative reference point and l must be equal to the

[14] Note that the difference between the locative temporal logic presented earlier [89] and the one presented in this chapter is the metrication: in [89], the author himself and Wim Koole presented a pure qualitative version of locative temporal logic whereas here metrication is applied to both the locative as well as the temporal sort.

value of η. Note that this formula is equivalent to $\bot$ (false) if η evaluates to 0.

- *Everywhere (at a distance)* ($\triangleq^{\mathsf{L}}_{=\eta}\ \varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where t is an arbitrary but fixed time point. The distance between the actual locative reference point and l must be equal to the value of η.

- *Always in the Past (at a distance)* ($\boxminus^{\mathsf{T}}_{=\tau}\ \varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where l is an arbitrary but fixed location and t is a time point that precedes the actual temporal reference point. The distance between the actual temporal reference point and t must be equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- *Always in the Future (at a distance)* ($\boxplus^{\mathsf{T}}_{=\tau}\ \varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where l is an arbitrary but fixed location and t is a time point that is preceded by the actual temporal reference point. The distance between the actual temporal reference point and t must be equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- *Everytime but Different (at a distance)* ($\triangle^{\mathsf{T}}_{=\tau}\ \varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where l is an arbitrary but fixed location and t is a time point that is different from the actual temporal reference point. The distance between the actual temporal reference point and t must be equal to the value of τ. Note that this formula is equivalent to $\bot$ (false) if τ evaluates to 0.

- *Everytime (at a distance)* ($\triangleq^{\mathsf{T}}_{=\tau}\ \varphi$) Formula φ is true at all pairs $\langle l, t\rangle$ where l is an arbitrary but fixed location. The distance between the actual temporal reference point and t must be equal to the value of τ.

Non-primitive metric locative temporal operators, e.g., dual and reflexive versions of above mentioned locative temporal operators can be defined in terms of the basic ones. We will give a list of the most important metric locative temporal operators in section 4.2.3 below. Note that we will provide in appendix A a deduction system for our two-sorted logic LTL that will be used throughout the whole part II. There, definitions of metric locative temporal operators will also be contained.

4.2.2 FORMAL SEMANTICS

As usual we give a Kripke style semantics for our locative temporal language, that is, we first introduce the notions of a locative temporal frame, a met-

ric locative temporal frame, and a metric locative temporal model and define afterwards the semantics of the well-formed formulae inductively on the complexity of their structure. Note that from the beginning on we will take into account two relations according to the two sorts of our locative temporal reference space (cf. also Koole [50]).

DEFINITION 4.4 (LOCATIVE TEMPORAL FRAME) Let L and T be non-empty sets of elements (*locations* and *time points*, respectively) where $L \cap T = \emptyset$. Furthermore, let $R_{\mathrm{L}}^{\approx}$ and $R_{\mathrm{T}}^{<}$ be binary relations on $L \times T$ (*locative* and *temporal accessibility*, respectively). Then, the structure $(L \times T, R_{\mathrm{L}}^{\approx}, R_{\mathrm{T}}^{<})$ is called a *locative temporal frame* or, for short, an *LT-frame*. $\Diamond$

In the sequel, we will presuppose the properties of our standard topology with locative temporal distance as presented in section 4.1.2. As with the temporal and locative cases alone, we shall make again no distinction between the alphabets (syntactic domains) $\mathcal{A}_{C_T}$ and $\mathcal{A}_{C_L}$ and the semantic domains $\mathbb{D}_{\mathrm{T}}$ and $\mathbb{D}_{\mathrm{L}}$, respectively.

DEFINITION 4.5 (METRIC LOCATIVE TEMPORAL FRAME) Let $\mathcal{F}_{\mathrm{LT}} = (L \times T, R_{\mathrm{L}}^{\approx}, R_{\mathrm{T}}^{<})$ be a locative temporal frame and let $\mathbb{D}_{\mathrm{L}}$ and $\mathbb{D}_{\mathrm{T}}$ be non-empty sets of elements (*locative* and *temporal metric domain*, respectively). Let $\rho_{\mathrm{L}} : L \times L \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a function (*locative distance function*) and let $\rho_{\mathrm{T}} : T \times T \longrightarrow \mathbb{D}_{\mathrm{T}}$ be a function (*temporal distance function*). Let $\oplus_{\mathrm{L}} : (\mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}}) \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a dyadic infix operator (*addition between locative distances*) and let $\oplus_{\mathrm{T}} : (\mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}}) \longrightarrow \mathbb{D}_{\mathrm{T}}$ be a dyadic infix operator (*addition between temporal distances*). Finally, let $0 \in \mathbb{D}_{\mathrm{L}}$ and $0 \in \mathbb{D}_{\mathrm{T}}$ be constants (*unity elements* w.r.t. $\oplus_{\mathrm{L}}$ and $\oplus_{\mathrm{T}}$, respectively). Then, the structure $(\mathcal{F}_{\mathrm{LT}}, \mathbb{D}_{\mathrm{L}}, \mathbb{D}_{\mathrm{T}}, \rho_{\mathrm{L}}, \rho_{\mathrm{T}}, \oplus_{\mathrm{L}}, \oplus_{\mathrm{T}}, 0)$ is called a *metric locative temporal frame* or, for short, a *metric LT-frame*. $\Diamond$

DEFINITION 4.6 (EVALUATION OF LOCATIVE TERMS) Let L be a set of locations and $l \in L$ and let T be a set of time points and $t \in T$. Let $\mathbb{D}_{\mathrm{L}}$ be a set of locative distances. Let Σ_{L} be a set of well-formed locative terms and let $\eta \in \Sigma_{\mathrm{L}}$. Furthermore, let $\alpha_{\mathrm{L}} : (L \times T) \times \Sigma_{\mathrm{L}} \longrightarrow \mathbb{D}_{\mathrm{L}}$ be a function (*locative binding*) assigning to each tuple of location and time point and a well-formed locative term a locative distance. Then, α_{L} is defined inductively

on the structure of locative terms in the following way:[15]

$$\alpha_{\mathrm{L}}(\langle l,t\rangle,\eta) = \begin{cases} d & \text{if } \eta = d \text{ and } d \in \mathcal{A}_{C_L} \\ d & \text{if } \eta = \zeta \text{ and } \zeta \in \mathcal{A}_{V_L} \\ & \quad \text{and } \zeta \in dom(\alpha_{\mathrm{L}}(\langle l,t\rangle)) \\ & \quad \text{and the value of } \zeta \text{ is } d \\ & \qquad \text{at position } \langle l,t\rangle \\ \alpha_{\mathrm{L}}(\langle l,t\rangle,\eta_1)) \oplus \alpha_{\mathrm{L}}(\langle l,t\rangle,\eta_2) & \text{if } \eta = \eta_1 \oplus \eta_2 \text{ is a compound} \\ & \quad \text{locative term} \end{cases}$$

If α_{L} has the property that $\alpha_{\mathrm{L}}(\langle l,t\rangle,\eta) = \alpha_{\mathrm{L}}(\langle l',t'\rangle,\eta)$ for all $\langle l,t\rangle, \langle l',t'\rangle \in (L \times T)$ and all terms η then we will call it again a *rigid locative binding.* $\diamond$

DEFINITION 4.7 (EVALUATION OF TEMPORAL TERMS) Let T be a set of time points and $t \in T$. Let $\mathbb{ID}_{\mathrm{T}}$ be a set of temporal distances. Let Σ_{T} be a set of well-formed temporal terms and let $\tau \in \Sigma_{\mathrm{T}}$. Furthermore, let $\alpha_{\mathrm{T}} : (L \times T) \times \Sigma_{\mathrm{T}} \longrightarrow \mathbb{ID}_{\mathrm{T}}$ be a function (*temporal binding*) assigning to each tuple of location and time point and a well-formed temporal term a temporal distance. Then, α_{T} is defined inductively on the structure of temporal terms in the following way:[16]

$$\alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) = \begin{cases} d & \text{if } \tau = d \text{ and } d \in \mathcal{A}_{C_T} \\ d & \text{if } \tau = \delta \text{ and } \delta \in \mathcal{A}_{V_T} \\ & \quad \text{and } \delta \in dom(\alpha_{\mathrm{T}}(\langle l,t\rangle)) \\ & \quad \text{and the value of } \delta \text{ is } d \\ & \qquad \text{at position } \langle l,t\rangle \\ \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau_1)) \oplus \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau_2) & \text{if } \tau = \tau_1 \oplus \tau_2 \text{ is a compound} \\ & \quad \text{temporal term} \end{cases}$$

If α_{T} has the property that $\alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) = \alpha_{\mathrm{T}}(\langle l',t'\rangle,\tau)$ for all $\langle l,t\rangle, \langle l',t'\rangle \in (L \times T)$ and all terms τ then we will call it again a *rigid temporal binding.* $\diamond$

DEFINITION 4.8 (METRIC LOCATIVE TEMPORAL MODEL) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{F}_{\mathrm{LT}}^{|\cdot|} = (\mathcal{F}_{\mathrm{LT}}, \mathbb{ID}_{\mathrm{L}}, \mathbb{ID}_{\mathrm{T}}, \rho_{\mathrm{L}}, \rho_{\mathrm{T}}, \oplus_{\mathrm{L}}, \oplus_{\mathrm{T}}, 0)$ be a metric locative temporal frame. Let α_{L} and α_{T} be a rigid locative and temporal binding, respectively, and let $\mathcal{I}_{\mathrm{LT}}$ be a function assigning to each propositional variable $p \in \mathcal{A}_P$ a subset of $L \times T$ on which p is true (*locative*

[15]Note that α_{L} now depends on a pair $\langle l,t\rangle$ which is different from the definition of a locative binding in chapter 3.

[16]Note that α_{T} now depends on a pair $\langle l,t\rangle$ which is different from the definition of a temporal binding in chapter 2.

temporal interpretation function), i.e., $\mathcal{I}_{\mathrm{LT}} : \mathcal{A}_P \longrightarrow \wp(L \times T)$. Then, the structure $(\mathcal{F}^{\parallel}_{\mathrm{LT}}, \alpha_{\mathrm{L}}, \alpha_{\mathrm{T}}, \mathcal{I}_{\mathrm{LT}})$ is called a *metric locative temporal model* or, for short, a *metric LT-model.* ◇

DEFINITION 4.9 (LOCATIVE TEMPORAL SATISFIABILITY) Let $\mathcal{A}_P$ be a set of propositional variables and let $\mathcal{M}^{\parallel}_{\mathrm{LT}} = (\mathcal{F}^{\parallel}_{\mathrm{LT}}, \alpha_{\mathrm{L}}, \alpha_{\mathrm{T}}, \mathcal{I}_{\mathrm{LT}})$ be a metric locative temporal model and let $\langle l, t\rangle \in (L \times T)$. Then, a well-formed formula φ *holds (is satisfied) in a metric locative temporal model* $\mathcal{M}^{\parallel}_{\mathrm{LT}}$ *at the point* $\langle l, t\rangle$, notation $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \varphi$, is defined inductively as follows:

(1) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \not\models_{\mathrm{LTL}} \bot$

(2) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} p$ iff $\langle l, t\rangle \in \mathcal{I}_{\mathrm{LT}}(p)$ for $p \in \mathcal{A}_P$

(3) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \varphi_1 \rightarrow \varphi_2$ iff if $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \varphi_1$ then $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \varphi_2$

(4) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \eta_1 = \eta_2$ iff $\alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_1) = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_2)$

(5) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \eta_1 < \eta_2$ iff $\alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_1) < \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_2)$

(6) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \tau_1 = \tau_2$ iff $\alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_1) = \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_2)$

(7) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \tau_1 < \tau_2$ iff $\alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_1) < \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_2)$

(8) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \varphi$ iff for all $\langle l', t\rangle \in (L \times T)$: if $R^{\approx}_{\mathrm{L}}\ \langle l, t\rangle\ \langle l', t\rangle$ and $\rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}((l, t), \eta)$ then $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi$

(9) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \triangle^{\mathrm{L}}_{=\eta} \varphi$ iff $0 \neq \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta)$ and for all $\langle l', t\rangle \in (L \times T)$: if $\langle l, t\rangle \neq_{\mathrm{L}} \langle l', t\rangle$ and $\rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta)$ then $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi$

(10) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \triangleq^{\mathrm{L}}_{=\eta} \varphi$ iff for all $\langle l', t\rangle \in (L \times T)$: if $R^{\mathrm{U}}_{\mathrm{L}}\ \langle l, t\rangle\ \langle l', t\rangle$ and $\rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta)$ then $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi$

(11) $\mathcal{M}^{\parallel}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \boxminus^{\mathrm{T}}_{=\tau} \varphi$ iff $0 \neq \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau)$ and for all $\langle l, t'\rangle \in (L \times T)$:

$$
\begin{array}{lll}
 & & \text{if } R^{\leq}_{\mathrm{T}}\ \langle l,t'\rangle\ \langle l,t\rangle \\
 & & \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{then } \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \\
(12) \quad \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t\rangle \models_{\mathrm{LTL}} \boxplus^{\mathrm{T}}_{=\tau}\varphi & \text{iff} & 0 \neq \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{and} \\
 & & \text{for all } \langle l,t'\rangle \in (L\times T): \\
 & & \text{if } R^{\leq}_{\mathrm{T}}\ \langle l,t\rangle\ \langle l,t'\rangle \\
 & & \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{then } \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \\
(13) \quad \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t\rangle \models_{\mathrm{LTL}} \triangle^{\mathrm{T}}_{=\tau}\varphi & \text{iff} & 0 \neq \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{and} \\
 & & \text{for all } \langle l,t'\rangle \in (L\times T): \\
 & & \text{if } \langle l,t\rangle \neq_{\mathrm{LT}} \langle l,t'\rangle \\
 & & \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{then } \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \\
(14) \quad \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t\rangle \models_{\mathrm{LTL}} \triangleq^{\mathrm{T}}_{=\tau}\varphi & \text{iff} & \text{for all } \langle l,t'\rangle \in (L\times T): \\
 & & \text{if } R^{\mathrm{U}}_{\mathrm{T}}\ \langle l,t'\rangle\ \langle l,t\rangle \\
 & & \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle,\tau) \\
 & & \text{then } \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \\
(15) \quad \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t\rangle \models_{\mathrm{LTL}} \forall\zeta:\varphi & \text{iff} & \text{for all } \alpha'_{\mathrm{L}} \text{ different} \\
 & & \text{from } \alpha_{\mathrm{L}} \text{ just on } \zeta: \\
 & & \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}],\langle l,t\rangle \models_{\mathrm{LTL}} \varphi \\
(16) \quad \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}},\langle l,t\rangle \models_{\mathrm{LTL}} \forall\delta:\varphi & \text{iff} & \text{for all bindings } \alpha'_{\mathrm{T}} \text{ different} \\
 & & \text{from } \alpha_{\mathrm{T}} \text{ just on } \delta: \\
 & & \mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}}[\alpha'_{\mathrm{T}}/\alpha_{\mathrm{T}}],\langle l,t\rangle \models_{\mathrm{LTL}} \varphi
\end{array}
$$

Thereby, $R^{\mathrm{U}}_{\mathrm{L}}$ is the universal relation on L, $R^{\mathrm{U}}_{\mathrm{T}}$ is the universal relation on T, and $\mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}]$ results from $\mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}}$ by replacing α_{L} with the new binding α'_{L}, i.e., $\mathcal{M}^{\downarrow\downarrow}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}] = (\mathcal{F}^{\downarrow\downarrow}_{\mathrm{LT}},\alpha'_{\mathrm{L}},\alpha_{\mathrm{T}},\mathcal{I}_{\mathrm{LT}})$. Substitution for the temporal binding is defined analogously.[17] ◇

Because of the two-sorted nature of our model, i.e., a locative temporal model we can define four kinds of validity depending on the fact whether a

[17] Note that the universal relations $R^{\mathrm{U}}_{\mathrm{L}}$ and $R^{\mathrm{U}}_{\mathrm{T}}$ have been introduced for reasons of analogy between modal operators and relations on the corresponding universes.

certain formula either holds for all locations or for all time points or for all pairs of locations and time points:[18]

DEFINITION 4.10 (LOCATIVE TEMPORAL VALIDITY) Let $\mathcal{M}^{||}_{LT}$ be a metric locative temporal model, i.e., $\mathcal{M}^{||}_{LT} = (\mathcal{F}^{||}_{LT}, \alpha_{LT}, \mathcal{I}_{LT})$.

1. A well-formed formula φ *is temporally valid in a metric locative temporal model* $\mathcal{M}^{||}_{LT}$ or, for short, *T-valid in* $\mathcal{M}^{||}_{LT}$, notation $\mathcal{M}^{||}_{LT}, l \models_{LTL} \varphi$, iff for all $t \in T : \mathcal{M}^{||}_{LT}, \langle l, t\rangle \models_{LTL} \varphi$ where $l \in L$ is arbitrary but fixed.[19]

2. A well-formed formula φ *is locatively valid in a metric locative temporal model* $\mathcal{M}^{||}_{LT}$ or, for short, *L-valid in* $\mathcal{M}^{||}_{LT}$, notation $\mathcal{M}^{||}_{LT}, t \models_{LTL} \varphi$, iff for all $l \in L : \mathcal{M}^{||}_{LT}, \langle l, t\rangle \models_{LTL} \varphi$ where $t \in T$ is arbitrary but fixed.[20]

3. A well-formed formula φ *is locatively and temporally valid in a metric locative temporal model* $\mathcal{M}^{||}_{LT}$ or, for short, *LT-valid in* $\mathcal{M}^{||}_{LT}$, notation $\mathcal{M}^{||}_{LT} \models_{LTL} \varphi$, iff for all $\langle l, t\rangle \in (L \times T) : \mathcal{M}^{||}_{LT}, \langle l, t\rangle \models_{LTL} \varphi$.

4. A well-formed formula φ *is valid*, notation $\models_{LTL} \varphi$, iff for all $\mathcal{M}^{||}_{LT} : \mathcal{M}^{||}_{LT} \models_{LTL} \varphi$.

$\Diamond$

4.2.3 DEDUCTION

Definitions of non-primitive locative and temporal operators as well as axioms and rules of locative and temporal logic can directly be adopted from the corresponding chapters. This is because not the syntax but the semantics of the operators differs, i.e., interpretation over a locative and temporal model there and interpretation over a two-sorted lovative temporal model here. For a succinct presentation in this section, we will avoid to repeat all such definitions, axioms, and rules. Instead we refer to chapters 2 and 3. But note that we are again not aiming here at a complete axiomatization of metric locative temporal logic.

In addition to the above, we have some more properties in locative temporal logic that combine locative and temporal operators (cf. also 4.1.2):

[18] Note that the four notions of validity will also give rise to a modified view on partial and total correctness of programs, e g , it would be possible to subdivide the classical notion of partial correctness into partial correctness w r t space and partial correctness w.r t time Extensive investigations would, in our view, be quite interesting but it must be postponed to future work because it goes far beyond the scope of this thesis

[19] Observe that T-validity conceptually coincides with temporal validity in chapter 2

[20] Observe that L-validity conceptually coincides with locative validity in chapter 3

1. $\vdash_{\mathrm{LTL}} \diamondsuit^{\mathrm{L}}\boxplus^{\mathrm{T}}p \rightarrow \boxplus^{\mathrm{T}}\diamondsuit^{\mathrm{L}}p$ (LT-Commutativity)

2. $\vdash_{\mathrm{LTL}} \diamondsuit^{\mathrm{T}}\boxdot^{\mathrm{L}}p \rightarrow \boxdot^{\mathrm{L}}\diamondsuit^{\mathrm{T}}p$ (TL-Commutativity)

3. $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}\boxplus^{\mathrm{T}}p \leftrightarrow \boxplus^{\mathrm{T}}\boxdot^{\mathrm{L}}p$ (LT-Confluency)

We have good reasons to assume that our locative temporal logic *without metrication* is both *sound*, i.e., if $\vdash_{\mathrm{LTL}} \varphi$ then $\models_{\mathrm{LTL}} \varphi$ for all LTL formulae φ and *complete*, i.e., if $\models_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \varphi$ for all LTL formulae φ w.r.t. the direct product of all models where the corresponding temporal relation is linear and the corresponding locative relation is an equivalence relation. This is because of work by Venema [87] and Stuhlmann-Laeisz [83] for the many-dimensional case, work by Koymans [54] and de Rijke [76] for the linear temporal case using the D-operator, and the fact that our locative logic is an S5-logic. We will be concerned with questions about soundness and completeness of the (non-metrical) logical system LTL in a joint work with Wiebe van der Hoek [91]. Soundness and (relative) completeness for the metrical case must be postponed to future work (cf. also Koymans [54]).

4.3 A RUNNING EXAMPLE (CONTD.)

Recall that we have started in the introduction a discussion about the well-known paradigm of dining philosophers at least for two reasons: firstly, we want to *illustrate* the principal distinctions between the various logics as introduced in this first part and, secondly, we want to *demonstrate* the adequacy of our specification method.

After having presented specifications of the dining philosophers using pure first-order predicate logic (cf. chapter 1), temporal logic (cf. chapter 2), and locative logic (cf. chapter 3) we are now going to give a locative temporal logic specification of the corresponding properties.

4.3.1 THE MODEL

An axiomatization of our model of the dining philosophers community will again comprise temporal properties as well as locative properties. But note that the properties are incorporated in the proof system of our locative temporal logic. What remains to be done in this section is, therefore, to provide the specification of behavioural properties of the dining philosophers community.

4.3.2 THE SPECIFICATION

For the specification of the requirements of the dining philosophers community we will need additional predicates:[21]

- ▷ *eaten*: has just eaten
- ▷ *lh*: possesses a fork in his left hand
- ▷ *rh*: possesses a fork in his right hand

Furthermore, let us introduce for notational convenience a constant predicate *bh* denoting the fact that a philosopher possesses forks in both hands:

$$bh \stackrel{\text{def}}{=} lh \wedge rh$$

Finally, the community of dining philosophers has to satisfy the following requirements:

EAT GUARANTEE All philosophers in the community will have eaten infinitely many times:

$$\boxdot^{\mathrm{L}}\boxplus^{\mathrm{T}}\diamondsuit^{\mathrm{T}}\, eaten$$

HAVING EATEN NEEDS FORKS A philosopher in the community can have eaten only if he possessed a fork in his left hand and a fork in his right hand at two successive points in time just before having eaten:

$$\boxdot^{\mathrm{L}}\boxplus^{\mathrm{T}}[\, eaten \rightarrow \diamondsuit^{\mathrm{T}}_{=1}\, (bh \wedge \diamondsuit^{\mathrm{T}}_{=1}\, bh)\,]$$

EXCLUSION OF FORK POSSESSION If some philosopher possesses two forks then his neighbours may not possess two forks at the same point in time:

$$\boxdot^{\mathrm{L}}\boxplus^{\mathrm{T}}[\, bh \rightarrow \neg\, \diamondsuit^{\mathrm{L}}_{=1}\, bh\,]$$

Note that we have again (cf. chapter 1) avoided to mention any start condition for the dining philosophers community.

[21] Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

4.4 SOME CONCLUSIONS

In this part, we have defined three different logics: temporal logic in chapter 2, locative logic in chapter 3, and locative temporal logic in chapter 4. Each of these logics can be regarded as a special kind of a modal logic. In temporal and locative logic only the modalities differ, that is, the interpretations of the corresponding modal operators in temporal logic is different from the one in locative logic. Temporal logic has been introduced to reason about time in terms of a temporal precedence relation on a set of time points whereas locative logic has been suggested to reason about space in terms of a locative reachability relation on a set of locations.

The third kind of modal logic, i.e., locative temporal logic can be regarded as an amalgamation of temporal and locative logic: the semantics of locative temporal logic has been constructed from the semantics of the latter two as the direct product of the temporal and locative universes together with suitable operators. The versions of the three logics have been chosen to be metrical in order to provide means for qualitative as well as quantitative reasoning about distributed real-time systems.

To illustrate the principal distinctions between the various logics and to demonstrate adequacy of our preferred specification method, i.e., locative temporal logic we have followed a running example throughout this first part. We began in chapter 1 with an informal discussion of the dining philosophers paradigm and presented there a formal specification of the corresponding properties in first-order predicate logic. All other three logics have also been applied to this paradigm in the corresponding chapters. Our version of the dining philosophers has been borrowed from Barringer et al. [6].

In the sequel, we will briefly compare the various specifications of the dining philosophers paradigm and we will draw some conclusions w.r.t. adequacy (cf. chapter 1), modelling, and structuring.[22]

4.4.1 ADEQUACY

Although, in this thesis, we shall not present any formal proof for adequacy, i.e., expressiveness and abstractness of LTL, we will give some indications for the following claim:

> When using LTL it is possible to make specifications of distributed real-time properties contain all relevant details but to avoid imple-

[22]Note that our presentation in tables 4.1, 4.2, and 4.3 to compare the several specifications of the dining philosophers paradigm has been inspired by the *square of languages* introduced by Koole in [50], page 8.

mentation bias.

For expressiveness we can state that all properties of the community of dining philosophers could be specified within LTL. Of course, such properties were very simple but they comprise different classes according to our classification in section 4.1.1: for instance, the property of eat guarantee (cf. table 4.1) constitutes a *distributed dynamic* property because it requires that every philosopher in the community must have eaten infinitely many times.

1PL	$\forall p, p' : [\ p \approx p' \rightarrow \forall t : \exists t' : t < t' \wedge eaten(p', t')\ ]$
TL	$\forall p, p' : [\ p \approx p' \rightarrow \boxplus^{\mathrm{T}} \oplus^{\mathrm{T}} eaten(p')\ ]$
LL	$\boxdot^{\mathrm{L}} [\ \forall t : \exists t' : t < t' \wedge eaten(t')\ ]$
LTL	$\boxdot^{\mathrm{L}} \boxplus^{\mathrm{T}} \oplus^{\mathrm{T}} eaten$

Table 4.1: Eat Guarantee

Another example is provided by the property of exclusion of fork possession between neighbours (cf. table 4.2). This also constitutes a *distributed dynamic* property because fork possession is regarded at the same point of time but at different philosophers.

1PL	$\forall p, p' : \forall t : [\ p \approx p' \wedge \lvert(p, p')\rvert_{\mathrm{L}} = 1 \rightarrow (bh(p, t) \rightarrow \neg bh(p', t))\ ]$
TL	$\forall p, p' : [\ p \approx p' \wedge \lvert(p, p')\rvert_{\mathrm{L}} = 1 \rightarrow \boxplus^{\mathrm{T}} (bh(p) \rightarrow \neg bh(p'))\ ]$
LL	$\boxdot^{\mathrm{L}} [\ \forall t : [\ bh(t) \rightarrow \neg \diamond^{\mathrm{L}}_{=1}\ bh(t)\ ]\ ]$
LTL	$\boxdot^{\mathrm{L}} \boxplus^{\mathrm{T}} [\ bh \rightarrow \neg \diamond^{\mathrm{L}}_{=1}\ bh\ ]$

Table 4.2: Exclusion of Fork Possession

From tables 4.1 and 4.2 we may also conclude that the other three approaches are expressive enough to specify the properties of the community of dining philosophers.

For abstractness we will compare our specification to the one presented by Barringer et al. [6] because we have borrowed the details from them. There, many different proposition symbols have been introduced, each one indexed by the "name" of the philosopher with which the corresponding property has to be associated. For example, $eaten_A$ and $eaten_B$ are two predicates denoting the fact that philosopher A and philospher B has just eaten, respectively. In our view, this is an implementation bias and, moreover, it is not really clear what relation exists, if at all, between such philosophers. For example, extending the set of philosophers by $C, D, \ldots$, etc. requires to specify the corresponding properties for the new philosophers by separate formulae although the properties are equivalent. Only the naming is different. Tables 4.1 and 4.2 show that using predicates over a philosophers domain might resolve this problem to a certain extent. But, in our view, such a solution will neither be sufficiently abstract w.r.t. space.

In contrary, increasing or decreasing the number of philosophers will, in our approach, not entail any further specification activities. For instance, in table 4.1, the property of eat guarantee is contained as LTL formula in the last row. This property is independent of the actual (reference) philosopher and is valid for all philosophers. It is also independent of any maximum number of philosophers. Similar arguments hold for the interface constraints between neighboured philosophers. So adding philosophers to a particular community of philosophers would leave our LTL specification unchanged. Only changing such properties or adding properties would lead to modifications.

Hence, we may conclude that LTL is a promising approach for the specification of distributed real-time systems. It has been demonstrated that our two-sorted approach LTL is expressive and abstract w.r.t. several classes of properties of such systems. To give more persuasive power to our conclusions we refer to part II where LTL is applied to more complex problems in the field of distributed real-time systems.

4.4.2 MODELLING

Nowadays, models for distributed real-time systems most often contain some relevant aspects but neglect others that are also relevant. For example, in [20], assumptions about the underlying network are stated in an imprecise informal way but they are important for the verification of the corresponding programs. In such a context, proofs become less rigorous and less convincing and one has

to rely on the author's and reader's intuitions and insights into the system at issue.

Another example comprises different models for the analysis of one and the same system but of different aspects, e.g., one model for investigations on the functionality of a distributed real-time system and another model for its reliability (cf. [17]). Such kind of separation would give rise to the same problem as above: the proofs might be formal for the corresponding aspects but as long as no statements and proofs about the interface between the two models and their impacts on one another are also provided such proofs would again be less rigorous and less convincing and as such we have again to rely on the author's and reader's intuitions and insights into the system at issue.

But it is exactly the intention of formal methods to overcome dependencies on author's and reader's intuition and insight:

> Our *two-sorted model* makes a clear understanding of locative temporal properties of distributed real-time systems indispensable and their formalization absolutely necessary.

Our view on models for distributed real-time systems has been strongly influenced by general systems theory [48]: in design and analysis of such systems it will be necessary to investigate *structural* and *behavioural* properties. In this sense, we have included space and time into our two-sorted model so that we are enforced to think about both structures each time properties of a distributed real-time system are considered (cf. section 4.1.2). As long as we do not specify both structural and behavioural properties, our specifications will be invalid in most of the cases.

4.4.3 STRUCTURING

Structuring and hence readability of specifications mainly depends, in our view, on three aspects: firstly, the number of basic predicates, secondly, the number and types of parameters of basic predicates, and, thirdly, the structuring of the formulae themselves. In this sense, we mean:

> Our *two-sorted modal logic* gives more structure to specifications of distributed real-time systems and thus makes them more readable.

In our running example, the number of basic predicates is equal to three in all four approaches—thereby neglecting the locative and temporal predicates such as "="—but there is a distinction in the number of parameters and their types. As becomes obvious from table 4.3 where all three basic predicates are contained, only in the LTL case the basic predicates are constant and as such

the formula will be purely propositional. In all other cases, the predicates depend on at least one parameter.

1PL	$\forall p, p' : \forall t :$ $[\ p \approx p' \wedge eaten(p', t)$ $\rightarrow$ $(\exists t' : t' < t \wedge \lvert(t, t')\rvert_{\mathrm{T}} = 1 \wedge bh(p', t'))$ $\wedge$ $(\exists t'' : t'' < t \wedge \lvert(t, t'')\rvert_{\mathrm{T}} = 2 \wedge bh(p', t''))$ $]$
TL	$\forall p, p' : [\ p \approx p' \rightarrow \boxplus^{\mathrm{T}} [\ eaten(p') \rightarrow \diamondsuit^{\mathrm{T}}_{=1}\ (bh(p') \wedge \diamondsuit^{\mathrm{T}}_{=1}\ bh(p'))\ ]\ ]$
LL	$\boxdot^{\mathrm{L}}$ $[\ \forall t : eaten(t) \rightarrow$ $[\ \exists t' : t' < t \wedge \lvert(t, t')\rvert_{\mathrm{T}} = 1 \wedge bh(t')\ ]$ $\wedge$ $[\ \exists t'' : t'' < t \wedge \lvert(t, t'')\rvert_{\mathrm{T}} = 2 \wedge bh(t'')\ ]$ $]$
LTL	$\boxdot^{\mathrm{L}} \boxplus^{\mathrm{T}} [\ eaten \rightarrow \diamondsuit^{\mathrm{T}}_{=1}\ (bh \wedge \diamondsuit^{\mathrm{T}}_{=1}\ bh)\ ]$

Table 4.3: Having Eaten Needs Forks

Instead of parameterized predicates over a philosophers domain in the 1PL- and TL-formulae we also could have introduced as many propositional variables as philosophers are present in the system.[23] But then the specification would become dependent on the number of philosophers and we would get the unpleasant situation that many differently named variables are used to denote conceptually the same thing. Opposed to that LTL formulae will be intuitively simple and easily surveyed as becomes obvious from all of the preceding tables.

[23] In [6] (pages 15 and 16), many different propositional variables have been introduced to denote the fact that a particular philosopher has eaten. These variables differ in their index, which is the name (absolute reference) of one philosopher.

Part II

Applications: DRTS

Chapter 5

Towards a Formal Approach

Outline

In part I of this thesis, we have developed a locative temporal reference space together with a locative temporal logic for specifying and reasoning about properties of distributed real-time systems. Part II of this thesis will now be devoted to some applications from the field of such systems. Before doing so in subsequent chapters, we will at first formalize some fundamental concepts that have already been discussed informally in chapter 1.

This chapter contains the following sections: Section 5.1 gives a short introduction to this second part. Especially the examples will be set in relation. In section 5.2, we provide a formalization of the notion of a distributed real-time system and some related concepts. These notions will be illustrated by the paradigm of a distributed watchdog.

5.1 INTRODUCTION

Models for behaviour of distributed real-time systems can be divided basically into two kinds [73]: firstly, *causality-based* models where two states s_1 and s_2 are ordered in the way (s_1, s_2) if and only if because of the causal structure of the system at issue s_2 is causally dependent on s_1. Secondly, *time-observer-based* models where two states s_1 and s_2 are ordered in the way (s_1, s_2) if and only if because of the global temporal structure of the system at issue s_1 temporally precedes s_2.

According to this division we can classify our approach as being *space-and-time-observer-based* or, in other words, our model of distributed real-time system behaviour is based on observations made by an

EXTERNAL OBSERVER IN SPACE AND TIME.

Thus, two states s_1 and s_2 are ordered in the way (s_1, s_2) if and only if because of the locative temporal structure of the system at issue s_2 is locatively and temporally reachable from s_1.[1]

This delivers a kind of horizontal view on distributed real-time systems and we can use our locative temporal logic for specifying and reasoning on one level of abstraction. As far as more than one level is concerned (see below) we can also make use of our locative temporal logic (LTL). This time LTL can be used for specifying properties on the various levels and LTL's deduction relation can be used to prove that the lower level of abstraction implies the upper level, in other words, that the properties of the upper level will be contained in the properties of the lower level. This gives rise to a certain notion of refinement that is based on the two fundamental notions of formal methods, i.e., *defined notions* and *primitive notions* (cf. Beth [11]).[2]

Horizontal and vertical abstraction and the usage of our two-sorted modal logic for specifying and reasoning about the corresponding properties will be relevant throughout this whole part. In section 5.2, basic concepts of distributed real-time systems that have been introduced informally in chapter 1 will be formalized to set the context for subsequent chapters. The definitions will be illustrated by the paradigm of a distributed watchdog.

In chapter 6, we will discuss basic communication techniques. According to the two-sorted nature of our approach, it seems worthwhile to discriminate communication techniques w.r.t. to their impact mainly on space (*locative discrimination*) or mainly on time (*temporal discrimination*). We have found

[1] In all three cases above it will be possible, of course, to regard transitions or actions instead of or additionally to states.

[2] We will call our notion of refinement *specification refinement*.

in the literature the following communication techniques: locative discrimination leads us to point-to-point and diffusion-based communication techniques (see section 6.1). Temporal discrimination leads us to synchronous and asynchronous communication techniques (see section 6.2).

Finally in chapters 7 and 8, we will investigate two paradigms from the field of distributed real-time systems. The first paradigm comprises the problem of atomic broadcast (cf. Cristian et al. [20]). The second paradigm comprises the problem of processor group membership (cf. Cristian [19]). Both paradigms will be investigated w.r.t. their service, i.e., a service specification will be given and w.r.t. a class of protocols, i.e., a protocol specification will be given. Service and protocol specifications are then related according to our notion of specification refinement. Note that in all subsequent chapters of part II we shall assume the properties for space and time as provided in appendix A, that is, connected space and linear time.

The relation between the two paradigms, i.e., atomic broadcast and group membership and their role within the context of a whole distributed real-time system is illustrated in figure 5.1.

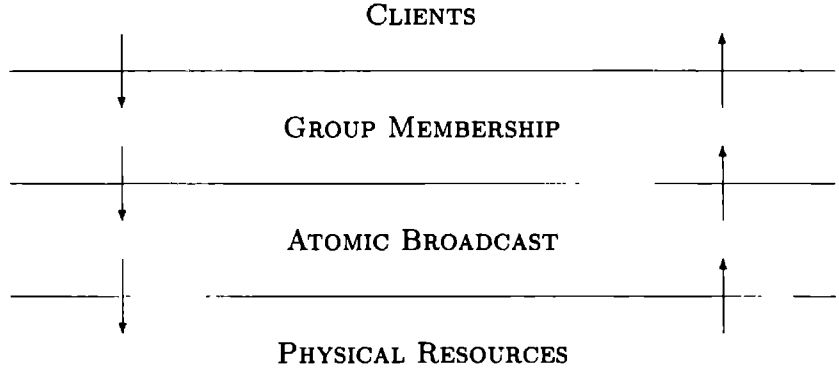


Figure 5.1: A Layered View on Distributed Real-Time Systems

The physical resources comprise the processors, channels, and clocks that have been associated with the notion of a process in chapter 1. The layered view above clarifies the interrelationships between the various levels. For example, clients will need information about the current state of the underlying network so that a request is issued to the corresponding group membership

server which notifies the requested information (cf. section 3.1.2). The group membership server itself makes use of an atomic broadcast server which takes care that information about the state of the underlying network can diffuse in the system. At the lowest level, the physical resources themselves are used to exchange data or signals between various processors in the system.

5.2 DEFINITIONS AND EXAMPLES

In the sequel, we shall need the notion of a locative reference space[3] that we have introduced formally in chapter 3 even though under the name "locative frame". So let us, at first, recall its definition:

> *A locative reference space is a structure* $(L, R_{\mathrm{L}}^{\approx})$ *where* L *is a non-empty set of locations and* $R_{\mathrm{L}}^{\approx}$ *is a binary relation of locative reachability on* L.

We will distinguish between a distributed real-time system and its configuration: the former is only an assembly of all significant components but does not relate them to each other and cannot show any behaviour. The latter additionally provides a relation between the components and behaviour can be associated with it.

DEFINITION 5.1 (DISTRIBUTED REAL-TIME SYSTEM) Let Λ be a set of sequential programs, Γ be a set of processors, Θ be a set of (physical) channels[4], and Ξ be a set of (hardware) clocks. A structure $(\Lambda, \Gamma, \Theta, \Xi)$ is called a *distributed real-time system* or shortly DRTS. ◇

In the next definition, it is important that Υ is a mapping from a set L of locations (meta-level) to the direct product of the set of programs, the set of processors, the powerset of channels, and set of clocks (object-level) and not vice versa.

DEFINITION 5.2 (CONFIGURATION OF A DRTS) Let $\mathcal{S} = (\Lambda, \Gamma, \Theta, \Xi)$ be a distributed real-time system and let $\mathcal{L} = (L, R_{\mathrm{L}}^{\approx})$ be a locative reference space. Let $\Upsilon : L \longrightarrow \Lambda \times \Gamma \times \wp(\Theta) \times \Xi$ be a function called *distribution function* of $\mathcal{S}$. A structure $(\mathcal{S}, \mathcal{L}, \Upsilon)$ is called a *configuration* of $\mathcal{S}$. ◇

[3]Note that in fact a locative temporal reference space will be needed. But for simplicity's sake and because we confine ourselves in this section to structural aspects only it seems justifiable.

[4]Our terminology differs here from common practice because our channels will usually be called "links". But we prefer to talk about physical and logical channels to make the abstraction level conspicuous.

In the sequel, we shall refer to the several components of the range of a distribution function in the following way:

- ▷ $prog(\Upsilon(l))$ refers to the program associated with l,
- ▷ $proc(\Upsilon(l))$ refers to the processor associated with l,
- ▷ $chan(\Upsilon(l))$ refers to the set of channels associated with l, and
- ▷ $clock(\Upsilon(l))$ refers to the clock associated with l.

EXAMPLE 5.1 (DISTRIBUTED WATCHDOG) A *distributed watchdog* [5] consists of $k \in \mathbb{N}$ processors P_i $(i = 1, \ldots, k)$ and a dedicated processor wd for the watchdog. The watchdog then wants to know whether the processors P_i are still functioning. It is usual to define a so called *watchdog process* to which processor wd is dedicated and to let this process periodically look whether the other processors are still alive. If this is not the case for at least one processor the watchdog has to set an alarm. The configuration of the network has been provided in figure 5.2.

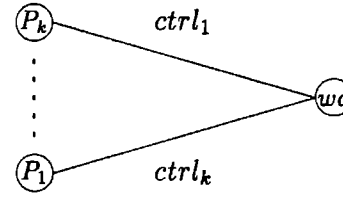


Figure 5.2: Distributed Watchdog

Instead of looking to the other processors—which can be quite complicated for the watchdog—one usually defines so called *signal* processes, one for each processor to be controlled, and requires that these processes periodically send a signal to the watchdog so that the watchdog can decide whether the others are alive.

A distributed watchdog can now be described as a particular configuration of a distributed real-time system. Let us denote this configuration by DW. Assuming $k = 4$, a sample configuration can be found in table 5.1 where Λ_{DW}

[5] See Hooman [43] for a thorough investigation of this problem. A distributed watchdog can be regarded as a much simpler version of a processor group membership (see chapter 8 for more details on the problem of group membership).

is the set of programs, Γ_{DW} is the set of processors, Θ_{DW} is the set of channels, Ξ_{DW} is the set of clocks, L is the set of locations, $R_L^{\approx}$ is the relation of locative reachability, and Υ_{DW} is the distribution function of DW.

L	$=$	$\{l_1, l_2, l_3, l_4, l_5\}$
$R_L^{\approx}$	$=$	$L \times L$
Λ_{DW}	$=$	$\{signal, watch\}$
Γ_{DW}	$=$	$\{P_1, P_2, P_3, P_4, wd\}$
Θ_{DW}	$=$	$\{ctrl_1, ctrl_2, ctrl_3, ctrl_4\}$
Ξ_{DW}	$=$	$\{clock\}$
$\Upsilon_{DW}(l_1)$	$=$	$(signal, P_1, \{ctrl_1\}, clock)$
$\Upsilon_{DW}(l_2)$	$=$	$(signal, P_2, \{ctrl_2\}, clock)$
$\Upsilon_{DW}(l_3)$	$=$	$(signal, P_3, \{ctrl_3\}, clock)$
$\Upsilon_{DW}(l_4)$	$=$	$(signal, P_4, \{ctrl_4\}, clock)$
$\Upsilon_{DW}(l_5)$	$=$	$(watch, wd, \{ctrl_1, ctrl_2, ctrl_3, ctrl_4\}, clock)$

Table 5.1: Configuration of DW

Note that we have not yet provided any behavioural properties of the components. For example, properties of the programs *watch* and *signal* have to be specified and also the fault-hypotheses of the various hardware components are still missing. On the other hand, we have taken some decisions in the early design process of a distributed watchdog, e.g., all processes can refer to the same central clock and each process has its own processor. ◇

As mentioned in chapter 1 we can determine several sub-systems. One such sub-system is given by a so called *communication network* constructed from the set of processors and the set of physical channels.

DEFINITION 5.3 (COMMUNICATION NETWORK) Let $\mathcal{S} = (\Lambda, \Gamma, \Theta, \Xi)$ be a distributed real-time system, let $\mathcal{L} = (L, R_L^{\approx})$ be a locative reference space, and let $\mathcal{C} = (\mathcal{S}, \mathcal{L}, \Upsilon)$ be a configuration of $\mathcal{S}$. Let $N : L \longrightarrow (\Gamma \times \wp(\Theta))$ be a function called *space-view function of* $\mathcal{C}$ with $N(l) \stackrel{\text{def}}{=} (proc(\Upsilon(l)), chan(\Upsilon(l)))$. The structure $(\mathcal{L}, \Gamma, \Theta, N)$ is called the *communication network* of $\mathcal{C}$. ◇

EXAMPLE 5.2 (DISTRIBUTED WATCHDOG (CONTD.)) The communication network of our distributed watchdog DW consists essentially of the set L of locations, the locative reachability relation $R_L^{\approx}$, the set Γ_{DW} of processors, the set Θ_{DW} of channels, and the space-view function $N_{DW} : L \longrightarrow (\Gamma_{DW} \times \wp(\Theta_{DW}))$ which is defined as $N_{DW}(l) = (proc(\Upsilon(l)), chan(\Upsilon(l)))$. The definitions are summarized in table 5.2.

L	$=$	$\{l_1, l_2, l_3, l_4, l_5\}$
$R_L^{\approx}$	$=$	$L \times L$
Γ_{DW}	$=$	$\{P_1, P_2, P_3, P_4, wd\}$
Θ_{DW}	$=$	$\{ctrl_1, ctrl_2, ctrl_3, ctrl_4\}$
$N_{DW}(l_1)$	$=$	$(P_1, \{ctrl_1\})$
$N_{DW}(l_2)$	$=$	$(P_2, \{ctrl_2\})$
$N_{DW}(l_3)$	$=$	$(P_3, \{ctrl_3\})$
$N_{DW}(l_4)$	$=$	$(P_4, \{ctrl_4\})$
$N_{DW}(l_5)$	$=$	$(wd, \{ctrl_1, ctrl_2, ctrl_3, ctrl_4\})$

Table 5.2: Communication Network of DW

Observe that the definitions are, of course, unchanged w.r.t. to the definition of the configuration of our distributed watchdog. This is because it is only a sub-system of the whole system. ◇

Another example for sub-systems in a configuration of a distributed real-time system is a *clock system* which is constructed from the set of clocks:

DEFINITION 5.4 (CLOCK SYSTEM) Let $\mathcal{S} = (\Lambda, \Gamma, \Theta, \Xi)$ be a distributed real-time system, let $\mathcal{L} = (L, R_L^{\approx})$ be a locative reference space, and let $\mathcal{C} = (\mathcal{S}, \mathcal{L}, \Upsilon)$ be a configuration of $\mathcal{S}$. Let $O : L \longrightarrow \Xi$ be a function called *clock distribution function* of $\mathcal{C}$ with $O(l) \stackrel{\text{def}}{=} clock(\Upsilon(l))$. A structure $(\mathcal{L}, \Xi, O)$ is called the *clock system* of $\mathcal{C}$. ◇

EXAMPLE 5.3 (DISTRIBUTED WATCHDOG (CONTD.)) The clock system of our distributed watchdog DW consists essentially of the set L of locations

that is unchanged w.r.t. its definition above, the locative reachability relation $R_L^{\approx}$ that is unchanged w.r.t. its definition above, the set Ξ_{DW} of clocks that is also unchanged w.r.t. its definition above, and the clock distribution function $O_{DW} : L_{DW} \longrightarrow \Xi_{DW}$ that is defined as $O_{DW}(l) = clock(\Upsilon(l))$. The definitions are summarized in table 5.3.

L	$=$	$\{l_1, l_2, l_3, l_4, l_5, l_6\}$
$R_L^{\approx}$	$=$	$L \times L$
Ξ_{DW}	$=$	$\{clock\}$
$O_{DW}(l_1)$	$=$	$clock$
$O_{DW}(l_2)$	$=$	$clock$
$O_{DW}(l_3)$	$=$	$clock$
$O_{DW}(l_4)$	$=$	$clock$
$O_{DW}(l_5)$	$=$	$clock$

Table 5.3: *DW* with a Central Clock System

Observe that the definitions are again unchanged w.r.t. to the definition of the configuration of our distributed watchdog. Moreover, the clock distribution function is very simple because a central clock system has been presupposed. ◇

Again, a locative reference space allows for separation between meta-level and object-level. Modifications to a distributed real-time system or its configuration or to particular sub-systems of its configuration can easily be described with the aid of such a reference space. For further clarification, let us decide to change the kind of clock system of the distributed watchdog above.

EXAMPLE 5.4 (DISTRIBUTED WATCHDOG (CONTD.)) Suppose that the old *central* clock system has to be exchanged with a *centrally controlled* clock system. The modifications of our distributed watchdog *DW* will be simple because they are restricted to the clock system only. The locative reference space would not change and also all other sub-systems remain unchanged. The

modified clock system Ξ' is provided in table 5.4 where the central clock *clock* is associated with the watchdog. All other locations are associated with their own local clock.

L	$=$	$\{l_1, l_2, l_3, l_4, l_5, l_6\}$
$R_L^{\approx}$	$=$	$L \times L$
Ξ'_{DW}	$=$	$\{c_1, c_2, c_3, c_4, clock\}$
$O_{DW}(l_1)$	$=$	c_1
$O_{DW}(l_2)$	$=$	c_2
$O_{DW}(l_3)$	$=$	c_3
$O_{DW}(l_4)$	$=$	c_4
$O_{DW}(l_5)$	$=$	$clock$

Table 5.4: *DW* with a Centrally Controlled Clock System

Obviously, the distribution function Υ_{DW} has also changed. This is easily derived from the clock distribution function because the modifications are restricted to this part of Υ_{DW}. ◇

We shall come back to the paradigm of a distributed watchdog in the next chapter where behavioural properties will be investigated. A complete specification will be given for structural as well as behavioural properties of a distributed watchdog.

Chapter 6

Basic Communication Techniques

Outline

According to the two-sorted nature of our approach, several communication techniques can be discriminated w.r.t. their impact mainly on space or mainly on time. In the literature, we have found w.r.t. locative discrimination *point-to-point-based* and *diffusion-based* communication techniques and w.r.t. temporal discrimination *synchronous* and *asynchronous* communication techniques.

This chapter contains the following sections: Section 6.1 investigates point-to-point- and diffusion-based communication techniques. Section 6.2 investigates synchronous and asynchronous communication techniques. In section 6.3, we draw some conclusions resulting from our investigations in this chapter.

6.1 POINT-TO-POINT VS. DIFFUSION-BASED COMMUNICATION

Locative discrimination of communication techniques comprises *point-to-point-based* and *diffusion-based* communication. In point-to-point-based communication the distance between sender (possibly more than one) and receiver (possibly more than one) is equal to one, that is, dissemination of messages is performed directly between two adjacent locations. Using *diffusion-based communication* dissemination of messages is considered in the whole communication network. The distance between sender (possibly more than one) and receiver (possibly more than one) can be any non-negative whole number including zero. In the latter case, sender and receiver coincide.

6.1.1 POINT-TO-POINT-BASED COMMUNICATION

In point-to-point-based communication techniques, the method of naming the communication partner, i.e., naming the receiving process by the sender or naming the sending process by the receiver essentially depends on the communication primitives supplied by the system. Thereby, *direct naming* is distinguished from *indirect naming* and *symmetric naming* from *asymmetric naming* (cf. Sloman and Kramer [82]). Direct naming means that those processes with which communication is to be performed must be named in the primitive. As a consequence every process needs to know the names of all processes to be communicated with. This is in contrast to the *principle of compositionality.* This principle is maintained when using indirect naming, e.g., through local port names or local channel names. Symmetric naming requires that both communication partners name one another whereas in asymmetric naming this will not be needed.

In locative temporal logic, explicit referencing to locations is not a priori provided neither for process names nor for channel names. In [90], we have indicated how to resolve this shortcoming: similar to the metrication of modal operators we have extended there our locative operators by some index expression. The index expression denotes a channel name so that not only reachability (in the sense of Kripke models) between two locations is required but also the channel names must be equal. This extension is similar to dynamic logic [38] in the sense that our channel names coincide with the transition labels of dynamic logic.[1] A second always possible solution will be followed here: we will introduce some fixed—in the sense of Garson [33]—channel domain and

[1] This similarity has been pronounced to us by Wim Koole.

predicates over that domain.

Depending on the number of locations point-to-point communication can be classified into the following patterns (cf. Sloman and Kramer [82]):

1. *One-to-One Communication*: a communication action takes place between a sender S and a receiver R using a particular transmission channel c (cf. figure 6.1).

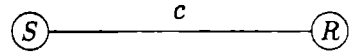


Figure 6.1: One Sender – One Receiver

The communication primitives of CSP such as $c!e$ (send value of expression e through channel c) and $c?x$ (receive data from channel c on variable x) may serve as examples for one-to-one communication.

2. *One-to-Many Communication*: a communication action takes place between a sender S and many receivers $R_1, \ldots, R_n$ using particular transmission channels $c_1, \ldots, c_n$ (cf. figure 6.2).

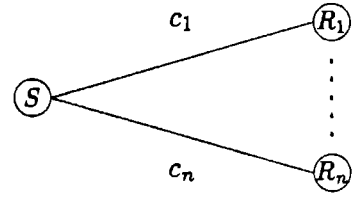


Figure 6.2: One Sender – Many Receivers

An example might be when a particular process wants to inform all other processes in a completely connected network about its own status then that process will send a message to all other processes, thereby, making use of the underlying communication network.

3. *Many-to-One Communication*: a communication action takes place between many senders $S_1, \ldots, S_n$ and one receiver R using particular transmission channels $c_1, \ldots, c_n$ (cf. figure 6.3).

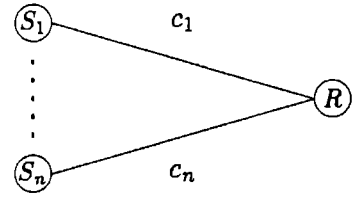


Figure 6.3: Many Senders – One Receiver

Recall our example of a distributed watchdog. There we had many processes that had to inform one watchdog whether they themselves were still alive. Such configuration makes use of a structure illustrated in figure 6.3. We shall come back to this example in a few minutes.

4. *Many-to-Many Communication*: a communication action takes place between many senders $S_1, \ldots, S_n$ and many receivers $R_1, \ldots, R_m$ and particular transmission channels $c_1^1, \ldots, c_n^m$ (cf. figure 6.4).

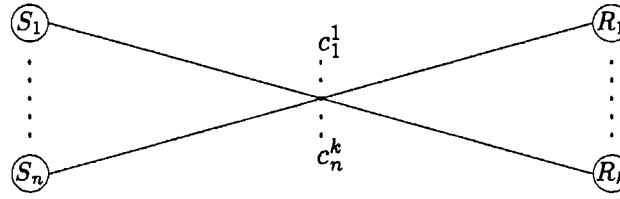


Figure 6.4: Many Senders – Many Receivers

An example for this quite complex structure will be a multiple user and multiple server situation.

We are now going to introduce LTL formulae that can be used to characterize communication patterns like those discussed above. Thereby, we will assume some finite set Θ of (physical) channels together with two predicates on Θ:[2] (let, thereby, ϑ be a variable ranging over Θ)

[2]Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

- ▷ $start(\vartheta)$: is start location of channel ϑ
- ▷ $end(\vartheta)$: is end location of channel ϑ

The following properties are static but distributed, that is, at least two locations are involved in the corresponding formulae (cf. chapter 4).

EXISTENCE OF A START LOCATION OF A CHANNEL Every channel that has an end location will also have a start location:

$$end(\vartheta) \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\ start(\vartheta)$$

EXISTENCE OF AN END LOCATION OF A CHANNEL Every channel that has a start location will also have an end location:

$$start(\vartheta) \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\ end(\vartheta)$$

UNIQUENESS OF A START LOCATION OF A CHANNEL Every start location of a channel is unique in the network:

$$start(\vartheta) \rightarrow \neg\nabla^{\mathrm{L}}\, start(\vartheta)$$

UNIQUENESS OF AN END LOCATION OF A CHANNEL Every end location of a channel is unique in the network:

$$end(\vartheta) \rightarrow \neg\nabla^{\mathrm{L}}\, end(\vartheta)$$

ONLY ONE START LOCATION IN THE NETWORK There is only one location in the network that serves as a start location of a channel:

$$start(\vartheta) \rightarrow \neg\nabla^{\mathrm{L}}\, start(\vartheta')$$

ONLY ONE END LOCATION IN THE NETWORK There is only one location in the network that serves as an end location of a channel:

$$end(\vartheta) \rightarrow \neg\nabla^{\mathrm{L}}\, end(\vartheta')$$

Not all of these properties will always be needed. The choice depends on the structure of interest. As an example, let us recall the paradigm of a distributed watchdog and let us try to specify such a system in LTL. Thereby, we restrict our attention to the processors to be controlled and the watchdog itself and neglect the environment as a separate location.

EXAMPLE 6.1 (DISTRIBUTED WATCHDOG (CONTD.)) Recall from chapter 5 that a distributed watchdog system consists of $k \in \mathbb{N}$ processors which have to be controlled and one dedicated processor which wants to know whether the former processors are still functioning correctly. Usually one requires that if at least one of the k processors fails to produce an *alive*-signal for at least δ time units then the watchdog has to give an alarm after these δ time units. Obviously, an alarm has to be generated only if at least one of the processors has failed during the last δ time units. Moreover, an *alarm* must only be generated at the watchdog site.

Let C be a set of channels and let c and c' be variables ranging over C. Then we get the following set of LTL formulae as a high level specification of a distributed watchdog:

STRUCTURAL PROPERTIES The structural properties as given below characterize the system structure, i.e., how channels in C are related to the locations in our locative reference space. Observe that all of these properties will be distributed static properties.

EXISTENCE OF AN END LOCATION OF A CHANNEL Every channel that has a start location also has an end location:

$$start(c) \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\, end(c)$$

EXISTENCE OF A START LOCATION OF A CHANNEL Every channel that has an end location also has a start location:

$$end(c) \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\, start(c)$$

UNIQUENESS OF A START LOCATION OF A CHANNEL Every start location can uniquely be identified in the network:

$$start(c) \rightarrow \neg\triangledown^{\mathrm{L}}\, start(c)$$

UNIQUENESS OF AN END LOCATION OF A CHANNEL Every end location can uniquely be identified in the network:

$$end(c) \rightarrow \neg\triangledown^{\mathrm{L}}\, end(c)$$

ONLY ONE END LOCATION IN THE NETWORK There is only one location that serves as an end location, i.e., the watchdog:

$$end(c) \rightarrow \neg\triangledown^{\mathrm{L}}\, end(c')$$

BEHAVIOURAL PROPERTIES The behavioural properties as given below characterize the events required and allowed in a distributed watchdog system. Observe that all of these properties will be distributed dynamic properties.

LIVENESS If a processor is not alive and stays not alive for δ time units then there will eventually be given an alarm after δ time units:

$$\neg alive \wedge \boxplus^{\mathsf{T}}_{\leq \delta}\, \neg alive \rightarrow \diamondsuit^{\mathsf{L}}_{=1}\, \diamondsuit^{\mathsf{T}}_{=\delta}\, alarm$$

SAFETY An alarm is only given when during the last δ time units a processor has not been alive:

$$alarm \rightarrow \diamondsuit^{\mathsf{L}}_{=1}\, \boxminus^{\mathsf{T}}_{\leq \delta}\, \neg alive$$

RELATING STRUCTURE AND BEHAVIOUR The structural properties will be related to the behavioural properties by means of relating the corresponding predicates, that is, we have to relate the predicates on the channel domain with the predicates denoting events in the system. Observe that all of these properties here will be local static properties.

ALARM ONLY AT WATCHDOG An alarm will only be given at the end location, i.e., by the watchdog:

$$alarm \rightarrow end(c)$$

$$start(c) \rightarrow \neg alarm$$

WATCHDOG ALWAYS ALIVE The end location, i.e., the watchdog may not go down:

$$end(c) \rightarrow alive$$

$\diamondsuit$

6.1.2 DIFFUSION-BASED COMMUNICATION

Information diffusion is a communication technique for conveying information among processors in a distributed real-time system. Following Cristian et al. [20], it can be characterized as follows:

- When a correct processor learns a piece of information it propagates the information to its neighbours by sending messages to them.

▷ If a correct neighbour does not already know that piece of information, it in turn propagates the information to its neighbours by sending them messages.

If some information propagates among neighbours then it is guaranteed that information diffuses among the correct processors in the system provided there are no network partitions.[3] Such a diffusion principle will not be restricted to a certain system level—as might be given, for example, by one layer of the OSI Reference Model [46]—but it can be applied to each such level.

In general, information diffusion (i-diffusion) comprises four activities: firstly, information *originates* somewhere, that is, it will come into existence by representing it in some form depending on its producer. Secondly, information will be *encoded*, that is, the original representation of information will be transformed into a representation suitable for dissemination. Thirdly, what then is disseminated and made available to others in the system is a message containing the original information. And fourthly, messages containing information will be *decoded*, that is, the original information will be recovered after message diffusion. Hence, information diffusion can be summarized by the following equation:

$$(1) \qquad \textit{i-diffusion} = \textit{origination} + \textit{encoding} + \textit{m-diffusion} + \textit{decoding}$$

The first two activities, i.e., origination and encoding and the last activity, i.e., decoding are all local activities, that is, they are restricted to one location. Opposed to that, message diffusion is a distributed activity, that is, more than one location is involved.

EXAMPLE 6.2 (INFORMATION DIFFUSION) Suppose we have a distributed real-time system of which we know or presuppose that every location will take care that, if and when a message has been received, this message propagates within some finite amount of time to all neighboured locations.

Furthermore, suppose that at the location marked with a "•" information i originates and is encoded to a message $m(i)$. According to the diffusion principle, this location will make $m(i)$ available to all its neighbours, which we have marked with a "1". Every such neighbour will then do the same so that $m(i)$ will reach their neighbours, which we have marked with a "2" (see figure 6.5 for an illustration). Observe that this course of events will, in principle, proceed ad infinitum.

[3] In the remainder of this section, we shall abstract from any failures of networks, i.e., processors and channels (cf. chapter 5) and presuppose that we have an ideal environment where processors, channels, and clocks are always correct. This assumption will be weakend in subsequent chapters.

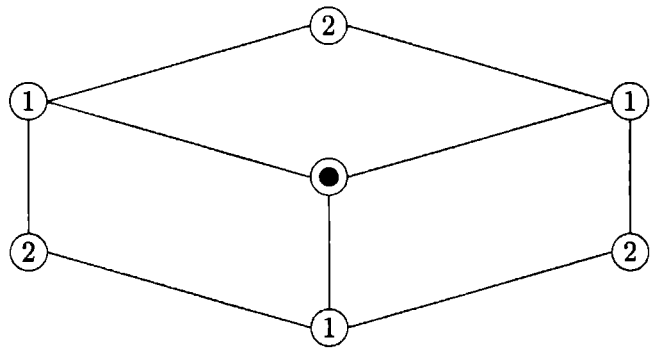


Figure 6.5: Information Diffusion

Hence, we can say that within a connected communication network information will be made available to all locations within some amount of time. This amount of time depends, in particular, on the time needed for message propagation between pairs of locations, on the time needed for encoding and decoding, on the time needed at each location before being able to propagate a message, and on the diameter of the network. In general, it can be assumed that the network is finite, i.e., the diameter of the network is finite so that information can diffuse within some finite amount of time. ◇

So, what really diffuses in information diffusion is some representation of information which is called here a message. All other components of information diffusion will be local to the corresponding locations. Therefore, proving properties about information diffusion will make use of properties about message diffusion. A basic result on message diffusion is contained in

THEOREM 6.1 (MESSAGE DIFFUSION) Let M be some message domain and m be a variable ranging over M. Let $\Re(m)$ be a predicate over M stating that *message m has been received.*[4] Finally, let $\varrho \in \mathbb{D}_T$ be some arbitrary but fixed temporal distance with $\varrho > 0$.

$$\forall \zeta : \; 0 < \zeta \rightarrow \; \Big[\; \circledast^{LT} \big[\Re(m) \rightarrow \boxdot^{L}_{=1} \diamondplus^{T}_{\leq \varrho} \Re(m) \big] \; \rightarrow \; \Big[\Re(m) \rightarrow \boxdot^{L}_{=\zeta} \diamondplus^{T}_{\leq (\zeta \otimes \varrho)} \Re(m) \Big] \; \Big]$$

[4]Note that $\Re(m)$ will be indefinite w.r.t. space and time (cf. section 4.1) and that $\mathbf{D}_L$ is assumed here to be the set of natural numbers including 0 (cf. appendix A), i.e., $\mathbf{D}_L = \mathbf{N}_0$.

◇

This theorem expresses that if at all reachable locations and time points one takes care that when a message has been received then all reachable neighbours of such a location will get that message not later than ϱ time units then when at the actual locative temporal reference point a message has been received then all reachable locations with distance ζ will get that message not later than $\zeta \otimes \varrho$ time units. This holds for all positive distances ζ.[5]

For notational convenience in the proof of Theorem 6.1, let us introduce the following abbreviation:

$$\widehat{\Re}(m) \stackrel{\text{def}}{=} \Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq \varrho} \Re(m) \tag{2}$$

PROOF 6.1 (THEOREM 6.1) The proof of the theorem will proceed inductively on ζ.

INDUCTIVE BASIS ($\zeta = 1$)

1. $\boxplus^{\mathrm{LT}} \widehat{\Re}(m)$	Assumption
2. $\Re(m)$	Assumption
3. $\boxdot^{\mathrm{L}} \boxplus^{\mathrm{T}} \widehat{\Re}(m)$	1., (LTL-71)
4. $\boxplus^{\mathrm{T}} \widehat{\Re}(m)$	3., (LTL-86), (LTL-136)
5. $\forall \delta : \boxplus^{\mathrm{T}}_{=\delta} \widehat{\Re}(m)$	4., (LTL-50)
6. $\forall \delta : (\delta = 0 \wedge \widehat{\Re}(m)) \vee \boxplus^{\mathrm{T}}_{=\delta} \widehat{\Re}(m)$	5., (LTL-51)
7. $\widehat{\Re}(m)$	6., (LTL-83), (LTL-136)
8. $\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq \varrho} \Re(m)$	7., (2)
9. $\boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq \varrho} \Re(m)$	8., 2., (LTL-136)

INDUCTIVE STEP ($\zeta > 1$)

We now have to prove that assuming the formula is valid for case ζ then it is also valid for case $\zeta + 1$:

10. $\boxplus^{\mathrm{LT}} \widehat{\Re}(m) \rightarrow [\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq \zeta \otimes \varrho} \Re(m)]$	Inductive Hypothesis
11. $\zeta > 1$	Assumption
12. $\varrho > 0$	Assumption

[5]Note that a more general version of this property has been provided with proposition A.43 in appendix A.

13. $\circledast^{LT}\,\widehat{\Re}(m)$ — Assumption
14. $\Re(m)$ — Assumption
15. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\Re(m)$ — 10., 13., 14., (LTL-136)
16. $\boxdot^{L}\circledast^{T}\,\widehat{\Re}(m)$ — 13., (LTL-71)
17. $\forall\zeta' : \boxdot^{L}_{=\zeta'}\,\circledast^{T}\,\widehat{\Re}(m)$ — 16., (LTL-9)
18. $\boxdot^{L}_{=\zeta}\,\circledast^{T}\,\widehat{\Re}(m)$ — 17., (LTL-83), (LTL-136)
19. $\boxdot^{L}_{=\zeta}\left[\forall\delta : \circledast^{T}_{=\delta}\,\widehat{\Re}(m)\right]$ — 18., (LTL-50)
20. $\boxdot^{L}_{=\zeta}\,\circledast^{T}_{=\zeta\otimes\varrho}\,\widehat{\Re}(m)$ — 19., (LTL-83), (LTL-136)
21. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{=\zeta\otimes\varrho}\,\widehat{\Re}(m)$ — 20., PROPOSITION A.18, (LTL-136)
22. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\widehat{\Re}(m)$ — 21., PROPOSITION A.29, (LTL-136)
23. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,[\Re(m) \rightarrow \boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)]$ — 22., (2)
24. $\boxdot^{L}_{=\zeta}\,[\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\Re(m) \rightarrow \diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)]$ — 23., PROPOSITION A.12, (LTL-136)
25. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\Re(m) \rightarrow \boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)$ — 24., (LTL-99), (LTL-136)
26. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\Re(m) \rightarrow \boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)$ — 25., PROPOSITION A.30, (LTL-136)
27. $\boxdot^{L}_{=\zeta}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)$ — 26., 15., (LTL-136)
28. $\boxdot^{L}_{=\zeta}\,\boxdot^{L}_{=1}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)$ — 27., (LTL-129), (LTL-136)
29. $\boxdot^{L}_{=\zeta\oplus 1}\,\diamondsuit^{T}_{\leq\zeta\otimes\varrho}\,\diamondsuit^{T}_{\leq\varrho}\,\Re(m)$ — 28., (LTL-100), (LTL-136)
30. $\boxdot^{L}_{=\zeta\oplus 1}\,\diamondsuit^{T}_{\leq(\zeta\otimes\varrho)\oplus\varrho}\,\Re(m)$ — 29., PROPOSITION A.24, (LTL-136)
31. $\boxdot^{L}_{=\zeta\oplus 1}\,\diamondsuit^{T}_{\leq(\zeta\oplus 1)\otimes\varrho}\,\Re(m)$ — 30., (LTL-77)

q.e.d.

Let us come back to the above example and demonstrate the use of the theorem, that is, how our intuition of message diffusion concides with the corresponding LTL-formulae.

EXAMPLE 6.3 (INFORMATION DIFFUSION (CONTD.)) Let us presuppose a set I of pieces of information and a set M of messages and two predicates:[6] (let, thereby, i and m be variables ranging over I and M, respectively)

[6]Note that the predicates are indefinite w.r.t. space and time (cf. section 4.1).

- ▷ *originated*(i): information i has originally been produced
- ▷ $\Re(m)$: message m has been received
- ▷ ϱ: time constant > 0 representing the *propagation time*, i.e., the maximum amount of time between reception of a message at one location and its reception at a neighboured location; thereby, we assume that no other time delay will have impact on message propagation
- ▷ d: space constant > 0 representing the *diameter* of the underlying network, i.e., the maximum number of hops for a message to disseminate in the whole network

Here, we are only interested in some liveness properties of a system S that makes use of the above mentioned diffusion technique in terms of the two predicates *originated*(i) and $\Re(m)$. Any other property, e.g., for encoding or decoding will be left open. Hence, we provide the following properties:

ORIGINATION Without any time delay, origination of information will lead locally to reception of the corresponding message:

$$\textit{originated}(i) \rightarrow \Re(m) \tag{3}$$

PROPAGATION Local reception of a message leads to propagation of that message to all neighbours:

$$\Re(m) \rightarrow \boxdot_{=1}^{\mathsf{L}} \diamondsuit_{\leq \varrho}^{\mathsf{T}} \Re(m) \tag{4}$$

The relation between piecess of information and messages will furtheron not be handled but suppose the following properties were given where we presuppose a function *representation* : $I \longrightarrow M$ delivering a representation of a piece of information (cf. also Koymans [54], page 95):

1. For all pieces of information there will be a corresponding message:

$$\forall i : \exists m : m = \textit{representation}(i)$$

2. The representation of information by messages will be unique:

$$\forall i : \forall m, m' : m = \textit{representation}(i) \wedge m' = \textit{representation}(i) \rightarrow m = m'$$

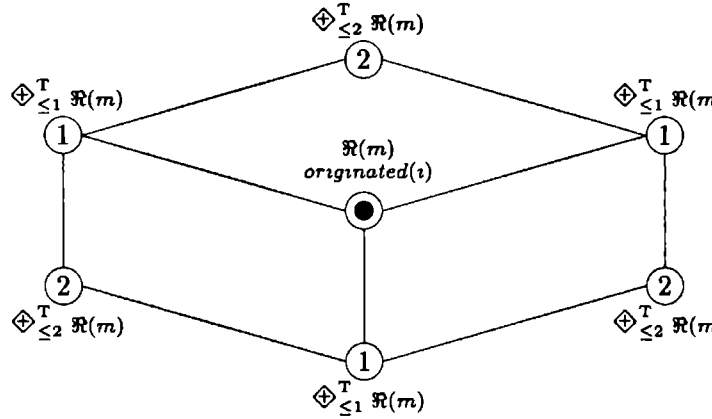


Figure 6.6: Information Diffusion indicated by LTL-Formulae

Now consider our picture from above (cf. figure 6.6) in a slightly modified version as presented in figure 6.6.[7] We have added some formulae to indicate how our reasoning in LTL is related to message diffusion in a distributed real-time system.

Suppose that information originates at the location marked with "•", i.e., the above property of origination becomes true now at that location. Then the corresponding formulae expressing reception of the corresponding message will become true at the various locations.

Without any further properties we can now easily prove the general liveness property that if information i has originally been produced then every location will have received the corresponding message m within $d \otimes \varrho$ time units:

PROPOSITION 6.1 Let Σ_S be the specification of S including the properties as given by the formulae (3) and (4) above.

$$\Sigma_S \vdash_{\mathrm{LTL}} \forall \zeta : 1 \leq \zeta \leq d \wedge originated(i) \rightarrow \boxdot^{\mathrm{L}}_{=\zeta}\, \Diamond^{\mathrm{T}}_{\leq \zeta \otimes \varrho}\, \Re(m)$$

$\Diamond$

PROOF 6.2 (PROPOSITION 6.1)

1. $1 \leq \zeta \leq d$ — Assumption

2. $originated(i)$ — Assumption

[7] For notational convenience in the picture, we have assumed a propagation time of 1, i.e., $\varrho = 1$.

3. $\Re(m)$	2., (3), (LTL-136)
4. $\circledast^{\mathrm{T}}_{=\delta}\,[\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq\varrho}\,\Re(m)]$	(4), PROPOSITION A.35
5. $\circledast^{\mathrm{T}}[\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq\varrho}\,\Re(m)]$	4., (LTL-137), (LTL-50)
6. $\boxdot^{\mathrm{L}}_{=\zeta}\,\circledast^{\mathrm{T}}\,[\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq\varrho}\,\Re(m)]$	5., (LTL-138)
7. $\boxdot^{\mathrm{L}}\circledast^{\mathrm{T}}\,[\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq\varrho}\,\Re(m)]$	6., (LTL-137), (LTL-9)
8. $\circledast^{\mathrm{LT}}[\Re(m) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq\varrho}\,\Re(m)]$	7., (LTL-71)
9. $\boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq\zeta\otimes\varrho}\,\Re(m)$	8., 1., 3., THEOREM 6.1, (LTL-136)

q.e.d.

With the theorem on *message diffusion* much of the work in proofs on *information diffusion* is left to derive local properties only, e.g., origination of information in this example. The local properties may well be static or dynamic (cf. chapter 4). For example, local properties may be provided for encoding of information and decoding of messages assuming that both activities will need some positive amount of time (cf. equation (1) above). $\diamondsuit$

6.2 SYNCHRONOUS VS. ASYNCHRONOUS COMMUNICATION

In general, three time periods during communication can be distinguished: an *establishment* phase where the communication partners come, if at all necessary, to an agreement about the properties of their communication, a *transaction* phase where the data is exchanged between the communication partners, and a *termination* phase where the communication partners come, if at all necessary, to an agreement about success of their communication (see figure 6.7 for an illustration).

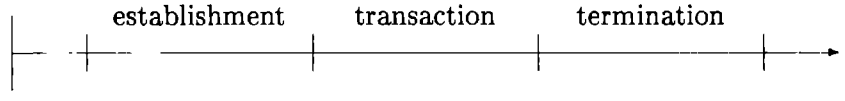


Figure 6.7: Communication Phases

Temporal discrimination of communication techniques comprises *synchronous* and *asynchronous* communication. They differ in their handling of the establishment and termination phase. The time point where, if at all necessary, agreement has been reached must not be finite but can be infinite. In the sequel, we will consider both techniques w.r.t. the three phase schema above.

6.2.1 SYNCHRONOUS COMMUNICATION

Synchronous communication is a communication technique which requires that sender and receiver reach agreement on successful communication at the same point in time. Synchronous communication can be divided w.r.t. time into two time periods (cf. also Hooman [43]): During the first period a process[8] establishes the connection to the communication partner. This will generally need some time before the corresponding partners have synchronized their activities, that is, the sender is willing to send and the receiver is willing to receive. The second period will then be needed to exchange the information between the two partners. At its end the partners immediately agree on success of their communication. This agreement will be implicit because send and reception take place simultaneously.

EXAMPLE 6.4 Suppose we have two communication partners, e.g., two processes π_1 and π_2. Before the two processes can exchange information it is necessary to synchronize their activities. Thus, we first have a synchronization period where at its end each process is to be indicating to the other its willingness to communicate. Immediately after this synchronization period both send and receive actions will be performed simultaneously on the corresponding processors ending in an agreement on successful communication (see figure 6.8 for an illustration). Without such simultaneity send and reception will not at all take place.

The time point of agreement is finite if all three periods are finite, i.e., the synchronization, transaction, and termination periods are all finite. Sometimes it is well allowed, e.g., by Hooman [43], that synchronization may take infinite time but that transaction and termination will both be finite. Taking into account our failure classification (cf. chapter 1) unsuccessful communication can be regarded as an omission failure, i.e., the time point of agreement is infinite. ◇

[8] According to our layered view, as discussed above, and assuming a one-to-one correspondence between processors and processes it will also be possible to read *processor* instead of *process*.

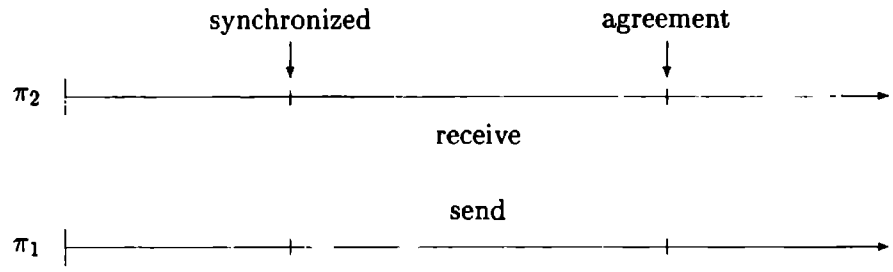


Figure 6.8: Synchronous Communication

As in the case of locative discrimination (see above), we are interested to specify synchronous communication properties by formulae that are in some sense characteristic, that is, where the structure of a formula indicates synchronicity.[9]

In [64], for example, synchronous communication is given in the programming language by a "pair of matching send and receive statements $l_s^s : \alpha \Leftarrow e : l_s^e$ and $l_r^s : \alpha \Rightarrow u : l_r^e$" where α denotes the channel through which communication should take place, e denotes a value that has to be sent, and u denotes a variable to which the received value has to be stored. Moreover, both statements are pre- and post-labelled by l_s^s and l_s^e and by l_r^s and l_r^e, respectively. The semantics is given by a joint transition, that is, both communication partners must be willing to communicate with each other and the value is immediately, i.e., without buffering, stored to the variable. Thus the characteristic feature of synchronous communication is manifested here in the buffer capacity, i.e., no buffering.

Using our two-sorted approach, we can directly observe matching send and receive actions in the specification of synchronous communication. Synchronicity can then be made explicit by using a predicate or it can be left implicit by requiring agreement on success of send and reception. In the sequel, we will provide formulae for both cases. Hence, let us introduce three predicates:[10]

- ▷ *sent*: a signal has been sent
- ▷ *received*: a signal has been received

[9] Recall that descriptional complexity and system complexity should be comparable. For example, it will be nice to have formulae characterizing synchronous communication on the one hand and asynchronous communication on the other hand (see also below).

[10] Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

▷ *synchronized*: has been synchronized

Let us, thereby, assume that only the fact of communication is interesting, that is, let us abstract from certain messages and from certain channels. According to our model of distributed real-time systems, properties can be divided into *distributed* and *local* properties:

DISTRIBUTED PROPERTIES are split into a sender's view and a receiver's view, that is, what has been sent must have been received and vice versa:

SENDER'S VIEW If a send action has been completed then there must also be some receive action that has been completed and the point in time must be the same:

$$sent \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\, received$$

RECEIVER'S VIEW If a receive action has been completed then there must also be a send action that has been completed and the point in time must be the same:

$$received \rightarrow \diamondsuit^{\mathrm{L}}_{=1}\, sent$$

LOCAL PROPERTIES will here take into account that before any transaction can take place there must be a synchronization period and that send and receive will exclude each other at the same point:

LOCAL LIVENESS Being synchronized leads to being sent and, therefore, to successful communication:

$$synchronized \rightarrow \diamondsuit^{\mathrm{T}}\, sent$$

NO SEND WITHOUT SYNCHRONIZATION A send action must be preceded by a synchronization action:

$$sent \rightarrow \diamondsuit^{\mathrm{T}}\, synchronized$$

NO SIMULTANEOUS SEND AND RECEIVE We also must prevent a system from simultaneously being sent and received:

$$\neg(sent \wedge received)$$

We have avoided to specify finite bounds in the formulae above. This is justified by the fact that we only wanted to stress the essential points. Of course, for the real-time case we will need time bounds (examples will be contained in the following chapters).

6.2.2 ASYNCHRONOUS COMMUNICATION

Asynchronous communication is a communication technique which is less reliable. An establishment period will not be needed and agreement on successful communication between the communication partners will not be required. Therefore, a user of such communication primitives is himself responsible that data reaches its destination, e.g., by certain protocols.

Asynchronous communication depends on some buffering capacity between the communication partners (see, e.g., Manna and Pnueli [64]). Sending data will only be possible if the buffer is not full and, complementary, receiving data can only take place if the buffer is not empty. Hence, asynchronous communication can be regarded as two times synchronous communication, namely, synchronous communication between a sender and a buffer and synchronous communication between that same buffer and a receiver.[11]

EXAMPLE 6.5 Suppose we have two communication partners, e.g., two processes π_1 and π_2 that need to exchange information. Using asynchronous communication primitives they do not need to synchronize their activities before starting to send or receive. They can start immediately when they want to communicate. The only restriction comes from the buffer that is realistically bounded. So, several send actions may occur consecutively before any receive action takes place (see figure 6.9 for an illustration).

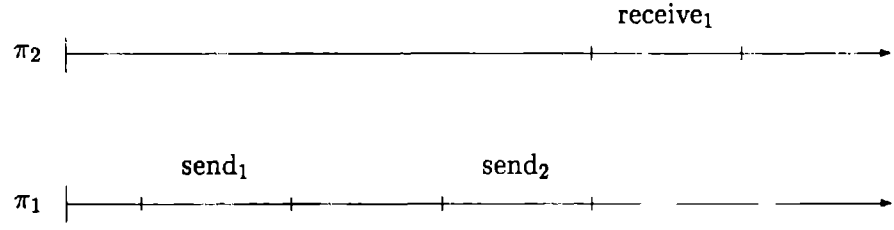


Figure 6.9: Asynchronous Communication

For the first send there exists the corresponding receive but for the second send success of communication is unpredictable from the figure above. ◇

As in the synchronous case above, we are again interested to specify asynchronous communication properties by formulae that are in some sense char-

[11] From an economical point of view, the buffer is an open market where goods are offered by the producer (sender) but where selling the goods by consumers (recievers) is left to chance.

acteristic, that is, where the structure of a formula indicates asynchronicity.[12]

In [64], for example, asynchronous communication is given in the programming language by send and receive statements $l_s^s : \alpha \Leftarrow e : l_s^e$ and $l_r^s : \alpha \Rightarrow u : l_r^e$" where α now denotes an bounded or unbounded buffer (channel) through which communication should take place, e denotes a value that has to be sent, and u denotes a variable to which the received value has to be stored. Moreover, both statements are again pre- and post-labelled by l_s^s and l_s^e and by l_r^s and l_r^e, respectively. The semantics is given by transitions that are dependent on the buffer state but that are independent of the communication partner. Thus the characteristic feature of synchronous communication is manifested here in the buffer capacity, i.e., bounded or unbounded buffering.

Using our two-sorted approach, similarity and difference between synchronous and asynchronous communication properties become directly apparent: in the synchronous case we made no use of temporal operators in the specification of distributed properties. Therefore, any buffering was disallowed. In the asynchronous case now, we will make use of temporal operators (see below). Hence, let us introduce the following predicates:[13]

- ▷ $started_s$: started to send a signal
- ▷ $started_r$: started to receive a signal
- ▷ *sent*: a signal has been sent
- ▷ *received*: a signal has been received
- ▷ *empty*: buffer is empty
- ▷ *full*: buffer is full

Let us, thereby, assume that only the fact of communication is interesting, that is, let us abstract from particular messages and from particular channels. Also changes in a particular buffer will not be considered. According to our model of distributed real-time systems, properties can again be divided into *distributed* and *local* properties:

DISTRIBUTED PROPERTIES allow for any time difference between sending and receiving and thus allowing for buffering. Providing some element for infinity, e.g., ∞, we can also include infinite buffering. Again we distinguish between sender's and reciver's properties:

[12]Recall that descriptional complexity and system complexity should be comparable. For instance, it will be nice to have formulae characterizing synchronous communication on the one hand and asynchronous communication on the other hand (see also above).

[13]Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

SENDER'S VIEW If a send action has been completed then there must also be some receive action that has been completed but the point in time may be different:

$$sent \rightarrow \diamondsuit^{\mathsf{T}}_{\leq c} \circledcirc^{\mathsf{L}}_{=1} received$$

where c is some fixed constant distance. If $c = \infty$ then it would be equivalent to $\top$ (true) and thus it expresses nothing.

RECEIVER'S VIEW If a receive action has been completed then there must also be a send action that has been completed but the point in time may be different:

$$received \rightarrow \diamondsuit^{\mathsf{T}} \circledcirc^{\mathsf{L}}_{=1} sent$$

Note the asymmetry in the indices between the sender's view and the receiver's view. This is because reception can only have taken place if there was some send.

It will also be possible to use a reflexive version of the temporal operators. Doing so would lead to a nice coincidence, namely, that in case send and receive would agree on success at the same time point it would be equivalent to synchronous communication.

LOCAL PROPERTIES will here take into account that the buffer used for communication may be full or empty in which case a send and a receive action must be disallowed, respectively. Additionally, let us assume a finite buffering capacity:

SEND LIVENESS Initiating a send action will lead to its completion if the corresponding buffer is not full:

$$started_s \wedge \neg full \rightarrow \diamondsuit^{\mathsf{T}} sent$$

RECEIVE LIVENESS Initiating a receive action will lead to its completion if the corresponding buffer is not empty:

$$started_r \wedge \neg empty \rightarrow \diamondsuit^{\mathsf{T}} received$$

SEND SAFETY Completion of a send action can only take place if it has been initiated:

$$sent \rightarrow \diamondsuit^{\mathsf{T}} started_s$$

RECEIVE SAFETY Completion of a receive action can only take place if it has been initiated:

$$receive \rightarrow \Diamond^{\mathrm{T}} started_r$$

NO SIMULTANEOUS SEND AND RECEIVE It will not be allowed that at the same point send and receive actions can be started:

$$\neg(started_s \wedge started_r)$$

ATOMIC SEND A send action is assumed to be atomic, that is, no other send or receive action can be started:

$$sent \rightarrow (\neg(started_s \vee started_r)\ \mathrm{S}\ started_s)$$

ATOMIC RECEIVE A receive action is assumed to be atomic, that is, no other receive or send action can be started:

$$received \rightarrow (\neg(started_s \vee started_r)\ \mathrm{S}\ started_r)$$

Observe that the real distinction between synchronous and asynchronous communication lies in the time difference between corrresponding send and receive actions, that is, in the asynchronous case the formulae of distributed properties contained temporal operators whereas in the synchronous case temporal operators were not present.

6.3 SOME CONCLUSIONS

From our investigations in section 6.1 it becomes obvious that point-to-point-based communication techniques require more specification effort than diffusion-based techniques. For example, structural properties characterizing the relation between locative reachability and physical channels have to be added to reason about the several communication patterns (see above). The properties of the locative temporal reference space are, in this sense, not sufficient for the specification of such communication techniques. This last observation, in fact, has led in [90] to another extension of LTL.

Opposed to point-to-point-based communication techniques a diffusion-based communication technique does not require for additional, purely structural properties (cf. section 6.1). Most of the specification effort can be summarized by a diffusion theorem that is suitable for the system at issue (see also chapters 7 and 8).

The reasons for such differences are founded, from our specification point of view, in the low abstraction level of point-to-point-based communication opposed to the higher level of abstraction of diffusion-based communication. Point-to-point-based communication can be used to implement diffusion-based communication. We may also conclude, that point-to-point-based techniques regard communication behaviour merely locally, e.g., most often only two processes are involved whereas diffusion-based techniques consider communication behaviour always system-wide.

Temporal discrimination has led, in section 6.2, to the investigation of synchronous and asynchronous communication techniques. Both communication techniques have been specified with particular predicates and without any special data structures. For example, in [64] the buffer capacity has been used to separate synchronous from asynchronous communication. In our LTL specifications, the discrimination between the two is manifested in different time points for the agreement on success of communication between sender and receiver. In sychronous communication, this time point is the same for both sender and receiver whereas in asynchronous communication this time point may be different.

Thus, specifications of synchronous and asynchronous communication in LTL will be free of implementation bias, i.e., particular data structures for buffers must not be incorporated in LTL (cf. also Koymans [54]). This is neither new nor surprising. What is new in LTL is the fact that the truth of the corresponding predicates and, therefore, of the formulae is dependent on the location. This allows to discriminate between the local properties of communication and the distributed ones. For example, whether a buffer is full or not has an impact on the local behaviour of a sender in asynchronous communication because it will not be allowed to send anything (see above).

Hence, we may conclude that our specification method LTL provides for means to syntactically characterize properties for senders and receivers in communication actions.

Chapter 7

Atomic Broadcast

Outline

A basic problem in distributed real-time systems is that of information dissemination among processes when the system or parts of it are sensitive of failures. A well-known example to overcome particular failure situations is an atomic broadcast protocol that provides a communication service to prevent a distributed real-time system from failing under certain conditions.

This chapter contains the following sections: Section 7.1 provides an informal discussion of the atomic broadcast service. In section 7.2, we present a formal specification of the atomic broadcast service. In section 7.3, we present the specification of protocols tolerating only particular processor failures, i.e., processor fail-stop without any network partitions. This class of protocols will be proved to satisfy the atomic broadcast service. In section 7.4, we draw some conclusions resulting from our investigations in this chapter.

7.1 PROBLEMATICAL NATURE

The problem of reaching consistent knowledge of the system state in a distributed real-time system is basically the problem of information diffusion (cf. section 6.1.2). Information that is owned at one processor has to be made available at all correct processors despite the occurrence of faults. To deal with certain faults occurring during execution a particular service will be needed: (Cristian et al. [20, 21][1])

> *An atomic broadcast service (ABS) is a communication service that enables the correct processors in a distributed real-time system to attain consistent knowledge of the system state, despite the occurrence of processor failure and random communication delays.*

The idea of an atomic broadcast service is to replicate system state information at several spatially dispersed processors and to use an *atomic broadcast protocol (ABP)* to disseminate state changes among so called *client processes* so that each one provides the same view on the system state.

An atomic broadcast service can be used to implement a *synchronous replicated storage*, i.e., a distributed and robust storage that provides the same contents at every correct processor in the system at any point in time except for some bounded time period of ϵ time units during which the corresponding protocol is executed.

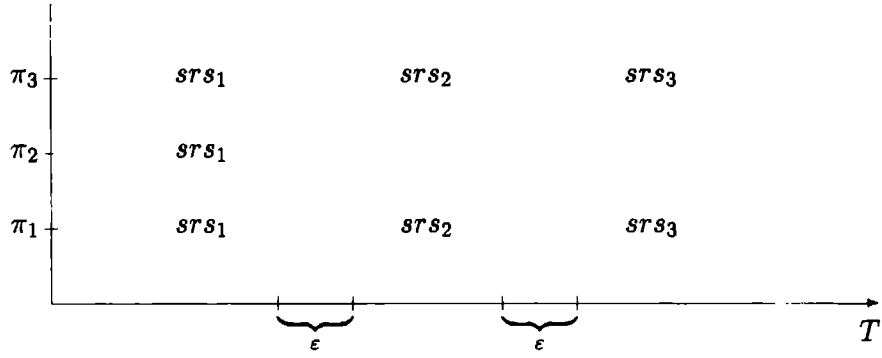
Although in a very informal manner Cristian et al. use graph theoretical means to model dynamically evolving networks. In our approach, too, graphs are used to model communication networks. But we use graphs in a static way only (cf. part I), i.e., for modelling the a priori known configuration of a distributed real-time system. The case that a network dynamically evolves in time will be modelled in our approach differently by making use of dynamically truth changing predicates. Hence, addition of new processors or connections must be taken into account from the beginning on when building the locative structure. We can then identify processors and channels that are not yet available in the system with not correctly functioning processors and channels, respectively.

EXAMPLE 7.1 Let π_1, π_2, and π_3 be some client processes. For simplicity, let us presuppose that each process has its own processor—recall that each

[1] [20] is a more detailed version of [21] that also provides verification proofs. However, in both papers the underlying communication network is not formalized and, therefore, no formal treatment of the network properties is possible.

processor is associated with a different location (cf. chapters 1 and 5). When correct each process wants to have a consistent and actual view of the system state. This state is contained in a synchronous replicated storage (srs) at each processor site.

Suppose that initially all processes have the same empty view on the system state, i.e., $srs_1 = \emptyset$ for π_1, π_2, and π_3 (cf. figure 7.1). It is assumed to be empty because nothing has happened before. After a certain time period the system state has changed, i.e., messages m_1, m_2, and m_3 have been generated. The time needed to disseminate such a change is bounded to ε time units as mentioned above so that after this time period a new view on the system state has been constructed and must be identical for all correct client processes, i.e., $srs_2 = \{m_1, m_2, m_3\}$. Furthermore, suppose that the processor of client π_2 has failed. Then, for π_1 and π_3 the knowledge of global system state will still be identical and nothing can be said about the view of π_2 (cf. figure 7.1).



Key: T: temporal reference space
$srs_1 = \emptyset$
$srs_2 = \{m_1, m_2, m_3\}$
$srs_3 = \{m_1, m_2, m_4\}$

Figure 7.1: Changes in Synchronous Replicated Storage

Now suppose a second change of system state has occurred after some time period and a third synchronous replicated storage $srs_3 = \{m_1, m_2, m_4\}$ has been generated by the corresponding atomic broadcast. Here message m_3 has been deleted from the synchronous replicated storage because, e.g., the information contained in m_3 has become too old indicated by some expiration time. At the same time a new message m_4 has been accepted, that is, m_4 will

be contained in srs_3. All remaining client processes must, therefore, have the same view on the new system state (cf. again figure 7.1).

In the period between the end of an atomic broadcast and the beginning of the next atomic broadcast the synchronous replicated storage will be unchanged for each client process. That is the knowledge of global system state is stable at each correct processor. Obviously, this is not true for the time period between the beginning of an atomic broadcast and its end. ◇

In [21], the following properties are mentioned to characterize an atomic broadcast service:

- ▷ *Atomicity*: if any correct processor delivers an update at time U on its clock, then that update was initiated by some processor and is delivered by each correct processor at time U on its clock.

- ▷ *Ordering*: all updates delivered by correct processors are delivered in the same order by each correct processor.

- ▷ *Termination*: every update whose broadcast is initiated by a correct processor at time T on its clock is delivered at all correct processors at time $T + \varepsilon$ on their clocks.

Termination ensures that every update is applied to the synchronous replicated storage by each correct processor ε time units after the initial broadcast. Atomicity ensures that every update is either applied by all correct processors or by none of them. And ordering ensures that all updates are applied in the same order by all correct processors.

As mentioned above, local clocks have been used for time stamping messages and computing deadlines. These clocks are assumed to be *approximately* synchronized (cf. original paper [21]). Opposed to that we shall abstract in the sequel from any problems around clock synchronization and assume instead a global clock or assume, which is equivalent, that all local clocks are *exactly* synchronized.

7.2 SERVICE SPECIFICATION

The specification of an atomic broadcast service will be given in terms of initiating message diffusion and delivery of such a message. On request a client process can then make use of such a service. An illustration of the environment of an atomic broadcast service (ABS) is provided in figure 7.2.

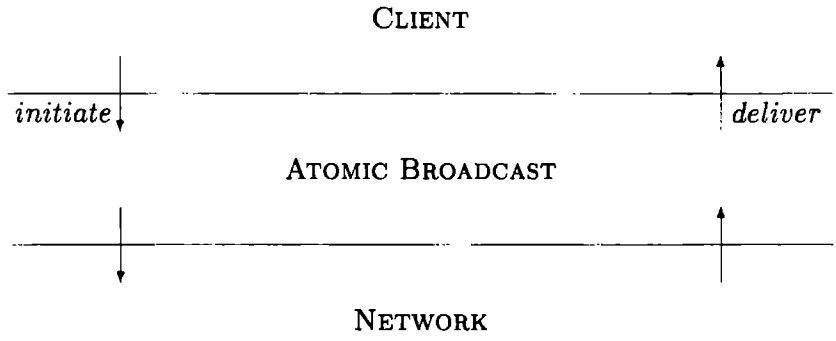


Figure 7.2: Atomic Broadcast Service

In the sequel, we will need for the service specification a finite set M of information items[2] together with several predicates:[3] (let, thereby, m, m' be variables ranging over M)

- ▷ *p-correct*: the associated processor is correct
- ▷ *ch-correct*: the associated channels are correct
- ▷ *c-correct*: the associated clock is correct
- ▷ *initiated*(m): diffusion of information m has been initiated
- ▷ *delivered*(m): information m has been delivered
- ▷ ε: time constant representing the maximum amount of time for information diffusion; it is called *diffusion time*

Because of our formal specification approach, i.e., locative temporal logic it will not be convenient to keep the division into atomicity, ordering, and

[2]In the sequel, we shall make no distinction between information items and messages although a clear separation would be more consequent (cf. section 6.1.2). Thus, we shall use both terms in an interchangable manner. Information diffusion and message diffusion will as such coincide (cf. equation 1 in section 6.1.2).

[3]Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

termination properties of the atomic broadcast service as introduced by Cristian et al. in [20]. Instead, we will use the terms *atomic liveness*, *safety*, and *ordering*. The translations from one terminology into the other is simple and we give hints in the context of the corresponding LTL formulae below. Moreover, we will extend the list of ABS properties with uniqueness requirements: according to Koymans [54], we will need unique identification for start and end of information diffusion.

ABBREVIATION LOCAL HW CORRECTNESS For notational convenience, let us introduce an abbreviation for the correctness of all hardware components associated with a location:

(1) $$correct \stackrel{\text{def}}{=} p\text{-}correct \wedge ch\text{-}correct \wedge c\text{-}correct$$

ATOMIC LIVENESS If the diffusion of a message has been initiated then that message will have been delivered at all locations not later than ϵ time units (provided that all is correct):

(2) $$correct \wedge initiated(m) \rightarrow \boxdot^{L} \diamondplus^{T}_{\leq \epsilon} \; [correct \rightarrow delivered(m)]$$

Note that our atomic liveness requirement comprises the old termination requirement and the second half of the old atomicity requirement.

SAFETY A message that has been delivered must have been initiated (provided that all is correct):

(3) $$correct \wedge delivered(m) \rightarrow \diamondminus^{LT} [correct \wedge initiated(m)]$$

Note that our safety requirement comprises the first half of the old atomicity requirement.

ORDERING Two messages that are delivered in a particular order must have been delivered in the same order at all locations (provided that all is correct):

(4) $$[(correct \wedge delivered(m)) \wedge \diamondminus^{T} (correct \wedge delivered(m'))]$$
$$\rightarrow$$
$$\boxdot^{L}[\neg \boxbslash^{T} [(correct \wedge delivered(m')) \wedge \diamondminus^{T} (correct \wedge delivered(m))]]$$

Note that our ordering requirement is essentially the same as the old one.

UNIQUENESS Diffusion of a message must be uniquely identifiable w.r.t. space and time (provided that all is correct). Message diffusion here is specified in terms of the predicates $initiated(m)$ and $delivered(m)$. Thus, we need the following uniqueness identifications:

$$(5) \qquad correct \wedge initiated(m) \rightarrow \neg\nabla^{L}\,[correct \wedge initiated(m)]$$

$$(6) \qquad correct \wedge initiated(m) \rightarrow \neg\nabla^{T}\,[correct \wedge initiated(m)]$$

$$(7) \qquad correct \wedge initiated(m) \rightarrow \neg\nabla^{LT}\,[correct \wedge initiated(m)]$$

$$(8) \qquad correct \wedge delivered(m) \rightarrow \neg\nabla^{T}\,[correct \wedge delivered(m)]$$

Observe that uniqueness w.r.t. initialization of information diffusion constitutes an assumption about the environment and can as such not be guaranteed by any protocol. Opposed to that uniqueness of delivery has to be guaranteed by a protocol.

In [20], unique identification of messages or of updates is not explicitly mentioned in the context of the ABS properties. Hence, it does not become clear enough whether unique identification of updates is assumed at this level (for a detailed discussion of this problem see Koymans [54], pages 79–80.).

7.3 A CLASS OF PROTOCOLS

In the introduction, we took the decision not to provide any programming language and to confine ourselves in this thesis to the specification of distributed real-time systems. Therefore, we shall only give a specification characterizing not exactly one particular protocol but a class of *atomic broadcast protocols* (ABP). Afterwards, we will formally prove that this class satisfies the properties of an atomic broadcast service.

The specification will be given in the following terms: information coming from a client forces the protocol to send a message to all neighbours of the current processor. Having received a message it will be delivered locally (cf. figure 7.3 for an illustration). Moreover, such a message will be propagated always and everywhere to the corresponding neighbours.[4]

[4]Note that expiration of messages as mentioned in the example of section 7.1 will not be part of the atomic broadcast protocol but of the maintenance of the synchronous replicated storage which itself is not considered here.

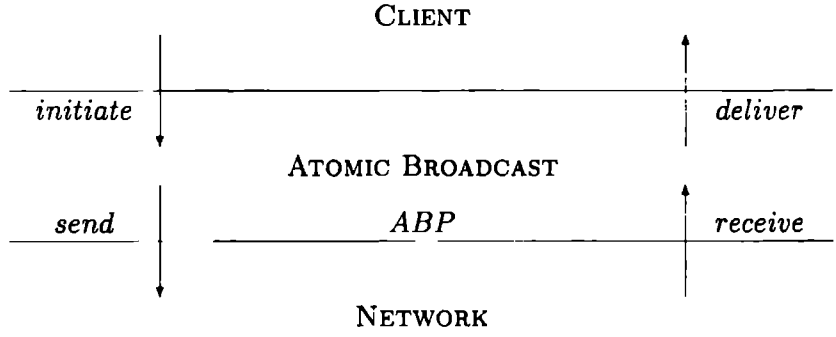


Figure 7.3: Atomic Broadcast Protocols

In addition to the set M of messages[5], the predicates, and abbreviations introduced in section 7.2, we will also make use of some new notions in the protocol specification:[6] (let, thereby, m, m' be variables ranging over M)

- $sent(m)$: message m has been sent
- $\Re(m)$: message m has been received
- ϱ: time constant > 0 representing the maximum amount of time for message propagation to neighbours; it is called *propagation time* of an atomic broadcast protocol
- $\mho$: space constant > 0 representing the *diameter* of the underlying network, that is, $\forall \zeta : 0 \leq \zeta \leq \mho$

7.3.1 FAULT-HYPOTHESES

Because distributed real-time systems are, in general, sensitive to failures we have to characterize those types of faults that may occur without any impacts on the systems' correct behaviour. This is usually done by stating possibly several distinct *fault-hypotheses (FH)* of the system or parts of it. According

[5]Note that we will make no distinction here between the information provided by a client and the message that contains the information for transmission.

[6]Note that the predicates will be indefinite w.r.t. space and time (cf. section 4.1).

to our view of processes and distributed real-time systems (cf. chapter 5), fault-hypotheses must be provided for clocks, channels, and the underlying physical network including the processors.

ABBREVIATION For notational convenience, let us introduce an abbreviation for the correctness of all network resources associated with a location:

(9) $$p\text{-}ch\text{-}correct \stackrel{\text{def}}{=} p\text{-}correct \wedge ch\text{-}correct$$

FH CLOCKS All clocks are assumed to be correct independent of the actual locative temporal reference point:

(10) $$c\text{-}correct$$

Although we know that clock synchronization is an essential problem in distributed real-time systems we leave it out here (see chapter 9) and presuppose a central clock system. We could also think, e.g., of a distributed clock system but then we would have to presuppose exactly synchronized clocks. Note that this differs from [20] in that there *approximately* synchronized clocks have been assumed.

FH CHANNELS All channels are assumed to be correct independent of the actual locative temporal reference point:

(11) $$ch\text{-}correct$$

FH PROCESSORS A processor that is incorrect will not send any message (*fail-silence* property):

(12) $$\neg p\text{-}correct \rightarrow \neg sent(m)$$

Additionally we will assume that an incorrect processor will remain to be incorrect (*fail-stop* property):

(13) $$\neg p\text{-}correct \rightarrow \boxplus^{\mathrm{T}} \neg p\text{-}correct$$

FH NETWORK PARTITIONING For all pairs of locations with a distance > 1 where the corresponding processor and set of channels are correct there must exist at least one intermediate location where the corresponding processor and set of channels are also correct:

(14) $$\forall \zeta : \Big[\; 1 < \zeta \wedge p\text{-}ch\text{-}correct \wedge \diamondsuit^{\mathrm{L}}_{=\zeta} \; p\text{-}ch\text{-}correct \;\Big] \rightarrow \Big[\; \exists \zeta' : 0 < \zeta' < \zeta \wedge \diamondsuit^{\mathrm{L}}_{=\zeta'} \; p\text{-}ch\text{-}correct \;\Big]$$

This formula characterizes the assumption of no partitions in a dynamically evolving physical network, as described in [20]. It follows that the network is at least 1-valent at all correct processors (cf. chapter 3).

7.3.2 PROTOCOL SPECIFICATION

In the sequel, we shall provide a characterization of behavioural properties of our class of atomic broadcast protocols. Two activities will be distinguished on this level (cf. figure 7.3): propagating information to all neighbours in the network by sending them messages and accepting information by receiving messages.

LOCAL PROPAGATION A message that has been sent will have been received locally without any time delay (provided that the local hardware resources are correct):

$$(15) \qquad correct \wedge sent(m) \rightarrow \Re(m)$$

PROPAGATION TO NEIGHBOURS A message that has been received will have been received by all neighbours within ϱ time units (provided that the local hardware resources are correct):

$$(16) \qquad correct \wedge \Re(m) \rightarrow \boxdot^{L}_{=1} \diamondsuit^{T}_{\leq \varrho} \; [correct \rightarrow \Re(m)]$$

SAFE PROPAGATION A message that has been received must have been sent by a processor previously (provided that the local hardware resources are correct):

$$(17) \qquad correct \wedge \Re(m) \rightarrow \Diamond^{LT} [correct \wedge sent(m)]$$

ORDERING The order of received messages must be the same at all processors (provided that the local hardware resources are correct):

$$(18) \qquad [correct \wedge \Re(m) \wedge \diamondsuit^{T} (correct \wedge \Re(m'))] \rightarrow \boxdot^{L} [\neg \boxtimes^{T} [(correct \wedge \Re(m')) \wedge \diamondsuit^{T} (correct \wedge \Re(m))]]$$

UNIQUENESS Unique identification of the messages transmitted on this level must be guaranteed. This has to be specified for the predicates $sent(m)$

and $\Re(m)$ in a suitable way (provided that the local hardware resources are correct):

$$(19) \qquad correct \wedge sent(m) \rightarrow \neg\nabla^{\mathrm{L}}\,[correct \wedge sent(m)]$$

$$(20) \qquad correct \wedge sent(m) \rightarrow \neg\nabla^{\mathrm{T}}\,[correct \wedge sent(m)]$$

$$(21) \qquad correct \wedge sent(m) \rightarrow \neg\nabla^{\mathrm{LT}}[correct \wedge sent(m)]$$

$$(22) \qquad correct \wedge \Re(m) \rightarrow \neg\nabla^{\mathrm{T}}\,[correct \wedge \Re(m)]$$

Note that only uniqueness w.r.t. reception can be guaranteed by the protocol and that uniqueness w.r.t. send must be provided by the environment (see figure 7.3 above).

Before being able to prove that the protocol specification is a specification refinement of the service specification we have to relate the corresponding primitive notions, that is, initialization of information diffusion on the higher level must be related to message send on the lower level and reception of a message on the lower level must be related to information delivery on the higher level.

INITIALIZATION ~ SEND If information diffusion has locally been initiated then the corresponding message must have been sent without any further time delay (provided that the local hardware resources are correct):

$$(23) \qquad correct \wedge initiated(m) \rightarrow sent(m)$$

If a message has been sent then information diffusion must have been initiated locally without any further time delay (provided that the local hardware resources are correct):

$$(24) \qquad correct \wedge sent(m) \rightarrow initiated(m)$$

DELIVERY ~ RECEPTION If information has locally been delivered then the corresponding message must have been received without any further time delay (provided that the local hardware resources are correct):

$$(25) \qquad correct \wedge delivered(m) \rightarrow \Re(m)$$

If a message has been received then the information must have been delivered locally without any further time delay (provided that the local hardware resources are correct):

$$(26) \qquad correct \wedge \Re(m) \rightarrow delivered(m)$$

ABS	*ABP*
– atomic liveness	– reception, propagation to neighbours, relative message diffusion
– delivery safety	– safe propagation
– delivery ordering	– reception ordering
– delivery uniqueness	– reception uniqueness
	– fault-hypotheses

Table 7.1: Relation between *ABS* and *ABP*

In table 7.1, we have set in relation the most important properties of the atomic broadcast service and the atomic broadcast protocol as defined above.

Thereby, we have avoided to make any distinction between information on the *ABS* level and messages on the protocol level. This could, when needed, easily be added to the above specification (see Koymans [54], example 3 of chapter 5).

7.3.3 CORRECTNESS

We now proceed to prove that the class of atomic broadcast protocols as defined by the specification formulae above is correct w.r.t. the specification of an atomic broadcast service as mentioned previously. That is, we will prove that the properties of *atomic liveness*, *safety*, and *ordering* and the corresponding uniqueness property can be derived from the properties of *ABP* provided that the clocks, channels, and processors meet their fault-hypotheses, respectively. This is stated in the following

THEOREM 7.1 Let *ABP* be a communication system implementing an atomic broadcast protocol satisfying the above protocol specification and let Σ_{ABP} be that specification. Furthermore, let $0 < \rho$ be the propagation time of the *ABP*, let $0 < \mho$ be the diameter of the underlying network, and let $FH(ABP)$ be the conjunction of all fault-hypotheses for *ABP*. Then, *ABP* satisfies the

properties of an atomic broadcast service, i.e.,[7]

$$\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow [\ \text{ATOMIC LIVENESS} \wedge \text{SAFETY} \wedge \text{ORDERING} \wedge \text{DELIVERY UNIQUENESS}\ ]$$

◇

Because of the nature of the theorem, that is, a conjunction as succedent of an implication, its proof will proceed in such a way that we first prove each conjunct separately and then gather the corresponding results so that the proof of the theorem becomes itself simple.

As already mentioned in section 6.1.2, we shall need for the proof of theorem 7.1 a special version of the message diffusion theorem: instead of the predicate $\Re(m)$ alone we have to take into account that the communication network is sensitive to errors, that is, send and reception of messages will depend on the correctness of the corresponding hardware resources. Thus, the corresponding properties become relative to such correctness.

THEOREM 7.2 (RELATIVE MESSAGE DIFFUSION) Let M be some message domain and m be a variable ranging over M and let $\varrho \in \mathbb{ID}_{\mathrm{T}}$ be some arbitrary but fixed and positive temporal distance.

$$\forall \zeta : 0 < \zeta \rightarrow [\ \circledast^{\mathrm{LT}}[(correct \wedge \Re(m)) \rightarrow \boxdot^{\mathrm{L}}_{=1} \diamondplus^{\mathrm{T}}_{\leq \varrho} (correct \rightarrow \Re(m))] \rightarrow \left[(correct \wedge \Re(m)) \rightarrow \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq (\zeta \otimes \varrho)} (correct \rightarrow \Re(m))\right]\ ]$$

◇

Observe that in the diffusion theorem above as well as in [20] it is not required that there really exists a location where a broadcasted message will have been received. All properties including the diffusion principle in [20] will be of relative nature, that is, they are relative w.r.t. the correctness of the underlying network. Therefore, the fault-hypothesis on network partitioning (see equation 14 above) will be applied as soon as the existence of such a location has to be taken into account.

[7] The names ATOMIC LIVENESS, SAFETY, ORDERING, and DELIVERY UNIQUENESS denote the equations 2, 3, 4, and 8, respectively.

Because theorem 7.2 is a slightly modified version of theorem 6.1 and its proof will be analogous to the proof of the latter we will avoid to give such a proof here.

LEMMA 7.1 (ATOMIC LIVENESS) Let Σ_{ABP} be the specification of an atomic broadcast protocol ABP satisfying the above specification and let $FH(ABP)$ be the conjunction of all fault-hypotheses for ABP. Furthermore, let $0 < \rho$ be the propagation time of the ABP and let $0 < \mho$ be the diameter of the underlying network.

$$\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow [\, correct \wedge initiated(m) \rightarrow \boxdot^{\text{L}} \diamondplus^{\text{T}}_{\leq \varepsilon} [correct \rightarrow delivered(m)]\,]$$

◇

For notational convenience in the proof of Lemma 7.1, let us introduce the following abbreviation:

$$\widehat{\Re}(m) \stackrel{\text{def}}{=} [correct \wedge \Re(m)] \rightarrow \boxdot^{\text{L}}_{=1} \diamondplus^{\text{T}}_{\leq \varrho} [correct \rightarrow \Re(m)] \tag{27}$$

PROOF 7.1 (LEMMA 7.1)

1. $\varrho > 0$ — Assumption
2. $\mho > 0$ — Assumption
3. $\forall \zeta : 0 \leq \zeta \leq \mho$ — Assumption
4. $correct \wedge initiated(m)$ — Assumption
5. $correct$ — 4., (LTL-80), (LTL-136)
6. $sent(m)$ — 4., (23), (LTL-136)
7. $\Re(m)$ — 6., 5., (15), (LTL-136)
8. $\boxplus^{\text{T}}_{=\delta} [(correct \wedge \Re(m)) \rightarrow \boxdot^{\text{L}}_{=1} \diamondplus^{\text{T}}_{\leq \varrho}(correct \rightarrow \Re(m))]$ — (16), PROPOSITION A.35
9. $\forall \delta : \boxplus^{\text{T}}_{=\delta} \widehat{\Re}(m)$ — (8), (27), (LTL-137)
10. $\boxplus^{\text{T}} \widehat{\Re}(m)$ — 9, (LTL-50)

11. $\forall\zeta : \boxdot^{\mathrm{L}}_{=\zeta} \circledast^{\mathrm{T}} \widehat{\Re}(m)$ 10., (LTL-138), (LTL-137)

12. $\circledast^{\mathrm{LT}} \widehat{\Re}(m)$ 11., (LTL-9), (LTL-71)

13. $\circledast^{\mathrm{LT}} \widehat{\Re}(m) \rightarrow \forall\zeta : [\mathit{correct} \wedge \Re(m)] \rightarrow \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq(\zeta\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$
27., Theorem 7.2, (LTL-82), (LTL-136)

14. $\forall\zeta : [\mathit{correct} \wedge \Re(m)] \rightarrow \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq(\zeta\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$
13., 12., (LTL-136)

15. $[\mathit{correct} \wedge \Re(m)] \rightarrow \forall\zeta : \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq(\zeta\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$
14., (LTL-82), (LTL-136)

16. $\forall\zeta : \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq(\zeta\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$ 15., 5., 7., (LTL-136)

17. $\forall\zeta : \boxdot^{\mathrm{L}}_{=\zeta} \diamondplus^{\mathrm{T}}_{\leq(\upsilon\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$
16., 3., Proposition A.25, (LTL-136)

18. $\boxdot^{\mathrm{L}} \diamondplus^{\mathrm{T}}_{\leq(\upsilon\otimes\varrho)} [\mathit{correct} \rightarrow \Re(m)]$ 17., (LTL-9)

19. $\boxdot^{\mathrm{L}} \diamondplus^{\mathrm{T}}_{\leq(\upsilon\otimes\varrho)} [\mathit{correct} \rightarrow \mathit{delivered}(m)]$ 18., (26), (LTL-136)

20. $\boxdot^{\mathrm{L}} \diamondplus^{\mathrm{T}}_{\leq\varepsilon} [\mathit{correct} \rightarrow \mathit{delivered}(m)]$ 19., Proposition A.25, (LTL-136)
provided that $\upsilon \otimes \varrho \leq \varepsilon$

q.e.d.

Lemma 7.2 (Safety) Let Σ_{ABP} be the specification of an atomic broadcast protocol ABP satisfying the above specification and let $FH(ABP)$ be the conjunction of all fault-hypotheses for ABP. Furthermore, let $0 < \rho$ be the propagation time of the ABP and let $0 < \upsilon$ be the diameter of the underlying network.

$$\Sigma_{ABP} \vdash_{\mathrm{LTL}} FH(ABP) \rightarrow [\, \mathit{correct} \wedge \mathit{delivered}(m) \rightarrow \diamondplus^{\mathrm{LT}} [\mathit{correct} \wedge \mathit{initiated}(m)] \,]$$

◇

Proof 7.2 (Lemma 7.2)

1. $\mathit{correct} \wedge \mathit{delivered}(m)$ Assumption

2. $correct$ — 1., (LTL-80), (LTL-136)
3. $\Re(m)$ — 1., (25), (LTL-136)
4. $\Diamond^{LT}[correct \wedge sent(m)]$ — 3., 2., (17), (LTL-136)
5. $\Diamond^{LT}[correct \wedge initiated(m)]$ — 4., (24), (LTL-80), (LTL-136)

q.e.d.

LEMMA 7.3 (ORDERING) Let Σ_{ABP} be the specification of an atomic broadcast protocol ABP satisfying the above specification and let $FH(ABP)$ be the conjunction of all fault-hypotheses for ABP. Furthermore, let $0 < \rho$ be the propagation time of the ABP and let $0 < \mho$ be the diameter of the underlying network.

$$\Sigma_{ABP} \vdash_{LTL} FH(ABP) \rightarrow [\,(correct \wedge delivered(m)) \wedge \Diamond^{T}(correct \wedge delivered(m'))\,] \rightarrow \boxdot^{L}[\,\neg\nabla^{T}[\,(correct \wedge delivered(m')) \wedge \Diamond^{T}(correct \wedge delivered(m))\,]\,]$$

◇

PROOF 7.3 (LEMMA 7.3)

1. $correct \wedge delivered(m) \wedge \Diamond^{T}(correct \wedge delivered(m'))$ — Assumption
2. $correct \wedge \Re(m) \wedge \Diamond^{T}(correct \wedge \Re(m'))$ — 1., (25), (LTL-80), (LTL-136)
3. $\boxdot^{L}[\neg\nabla^{T}[correct \wedge \Re(m') \wedge \Diamond^{T}(correct \wedge \Re(m))]]$ — 2., (18), (LTL-136)
4. $\boxdot^{L}[\neg\nabla^{T}[correct \wedge delivered(m') \wedge \Diamond^{T}(correct \wedge delivered(m))]]$ — 3., (26), (LTL-80), (LTL-136)

q.e.d.

LEMMA 7.4 (TIME UNIQUENESS) Let Σ_{ABP} be the specification of an atomic broadcast protocol ABP satisfying the above specification and let $FH(ABP)$ be the conjunction of all fault-hypotheses for ABP. Furthermore, let $0 < \rho$

be the propagation time of the ABP and let $0 < \mho$ be the diameter of the underlying network.

$$\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow [\, correct \wedge delivered(m) \rightarrow \neg\nabla^{\mho}\, [correct \wedge delivered(m)]\,]$$

$\Diamond$

Proof 7.4 (Lemma 7.4) We prove the lemma by contradiction, that is:

$$\Sigma_{ABP} \vdash_{\text{LTL}} \neg[\, FH(ABP) \rightarrow [\, correct \wedge delivered(m) \rightarrow \neg\nabla^{\mho}\, [correct \wedge delivered(m)]\,]\,] \rightarrow \bot$$

1. $FH(ABP) \wedge correct \wedge delivered(m) \wedge \nabla^{\mho}(correct \wedge delivered(m))$ Assumption
2. $correct \wedge delivered(m)$ 1., (LTL-80), (LTL-136)
3. $correct \wedge \Re(m)$ 2., (25), (LTL-80), (LTL-136)
4. $\neg\nabla^{\mho}\, [correct \wedge \Re(m)]$ 3., (22), (LTL-136)
5. $\nabla^{\mho}(correct \wedge delivered(m))$ 1., (LTL-80), (LTL-136)
6. $\nabla^{\mho}(correct \wedge \Re(m))$ 5., (25), (LTL-80), (LTL-136)
7. $\bot$ 4., 6.

Because the negation of the original formula leads to false the orginal formula itself must be true. *q.e.d.*

Proof 7.5 (Theorem 7.1) Recall that Σ_{ABP} is the specification of an atomic broadcast protocol ABP satisfying the above specification, that $0 < \rho$ is the propagation time of the ABP, that $0 < \mho$ is the diameter of the underlying network, and that $FH(ABP)$ is the conjunction of all fault-hypotheses for ABP.

Because we have proved

▷ $\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow$ ATOMIC LIVENESS Lemma 7.1

▷ $\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow$ SAFETY Lemma 7.2

▷ $\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow$ ORDERING Lemma 7.3

▷ $\Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow$ DELIVERY UNIQUENESS Lemma 7.4

and with the scheme

$$[(\varphi \rightarrow \varphi_1) \wedge (\varphi \rightarrow \varphi_2)] \leftrightarrow [\varphi \rightarrow (\varphi_1 \wedge \varphi_2)]$$

of propositional logic we can directly derive:

$$\begin{aligned} \Sigma_{ABP} \vdash_{\text{LTL}} FH(ABP) \rightarrow [& \text{ ATOMIC LIVENESS} \\ & \wedge \text{ SAFETY} \\ & \wedge \text{ ORDERING} \\ & \wedge \text{ DELIVERY UNIQUENESS} \\ &] \end{aligned}$$

q.e.d.

7.4 SOME CONCLUSIONS

Compared to the original work by Cristian et al. [20] we have presented more rigorous specifications for both the atomic broadcast service and a class of protocols. All static properties, whether local or distributed, have been left informal in [20]. For example, properties about the network such as the absence of network partitions have been provided there only informally. Hence, we may conclude that our proofs will be more rigorous than those given by Cristian et al. On the other hand, our protocol specification is much simpler and less details have been regarded.

Another distinction is given by the formal method applied: in [20]: it is first-order predicate logic for the specification of the service and a particular Pascal-like programming language used by Cristian et al. whereas we made use of our locative temporal logic for both the service specification as well as the protocol specification. Although the use of a particular programming language would be better suited for verification purposes a strict formal verification approach, e.g., a formal semantics of the language will also be missing in [20].

In [95], Ping Zhou and Jozef Hooman have also investigated the atomic broadcast problem as presented in [20]. They have provided service and protocol specifications in first-order predicate logic and have rigorously proved

that a protocol meeting their specification will also satify the properties of the atomic broadcast service. Compared to their work our presentation in this chapter differs not only in the applied formal method but it is also a much simpler version of a class of protocols.

So, we mean that this chapter has shown the usefulness of our two-sorted approach for the specification and verification of systems that are more complex than those discussed in previous chapters of this thesis. In particular, we have seen that most of the work in proving liveness properties can be subsumed under a suitable version of the message diffusion theorem. Moreover, it is important to take care that all ABS and ABP properties were of relative nature, that is, all these properties express nothing about the existence of locations *and* time points where a broadcasted message really has been received.

Chapter 8

Group Membership

Outline

Another basic problem in distributed real-time systems is that of reaching agreement on processor group membership, i.e., which processors are functioning correctly at a certain point in time. A well-known solution is a processor group membership protocol that guarantees the same view among the correct processors in the system despite the occurrence of processor startups or failures.

This chapter contains the following sections: Section 8.1 provides an informal discussion of the processor group membership service. In section 8.2, we present a formal specification of the processor group membership service. In section 8.3, we present a formal specification of a class of protocols based on periodic broadcast. This class of protocols will be proved to satisfy the group membership service. In section 8.4, we draw some conclusions resulting from our investigations in this chapter.

8.1 PROBLEMATICAL NATURE

The problem of reaching agreement on the view of processor group membership in a distributed real-time system is basically the problem of providing the same view on the presence and absence of processors at all correctly functioning processors. To deal with failing and (re-) starting processors and, therefore, with changes in the group membership a particular service is needed: (Cristian [19] and Cristian et al. [22][1])

> *A processor group membership service (GMS) is a distributed service that enables the correct processors in a distributed real-time system to provide the same view of present and absent processors and to maintain this view consistently.*

Changes in group membership either occur when a processor initially starts up or when a processor restarts after a failure has occurred or when a processor has failed and thus no longer functions correctly until restart. A further case can be thought of when new connections between already existing processors or new processors come into existence. All such events correspond to a so called *dynamically evolving network*.

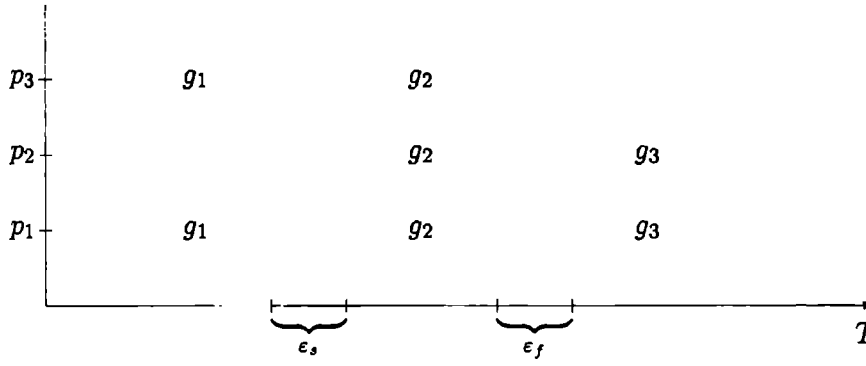
Although in a very informal manner, Cristian et al. [22] use graph theoretical means to model dynamically evolving networks. In our approach, too, graphs are used to model communication networks. But we use graphs in a static way only (cf. part I), i.e., for modelling the a priori known configuration of a distributed real-time system. The case that a network dynamically evolves in time will be modelled in our approach differently by making use of dynamically truth changing predicates. Processors and channels that are not yet available in the system can then be identified with not correctly functioning processors and channels, respectively.

EXAMPLE 8.1 Let p_1, p_2, and p_3 be some processors—recall that each processor is associated with a different location. Changes in group membership only occur if a processor starts or fails to deliver its service.

Suppose that initially processors p_1 and p_3 are functioning correctly and that they form a group, e.g., $g_1 = \{p_1, p_3\}$ (cf. figure 8.1). After some time period the configuration of correctly functioning processors changes because processor p_2 starts up execution where the other processors remain correct. The time needed to construct a group after the start of a processor is bounded

[1] [22] is a more detailed version of [19] and additionally provides a third protocol to implement the group membership service.

to ε_s time units so that after that time period a new group has been constructed and will be identical at all correct processors, e.g., $g_2 = \{p_1, p_2, p_3\}$ (cf. figure 8.1).



Key: T: temporal reference space
$g_1 = \{p_1, p_3\}$
$g_2 = \{p_1, p_2, p_3\}$
$g_3 = \{p_1, p_2\}$

Figure 8.1: Changes in Group Membership

Now suppose a change in the configuration of correctly functioning processors has occurred after some time period: processor p_3 has failed to deliver its service whereas the other two remain correct. The time needed to construct a group after the failure of a processor is bounded to ε_f time units so that after that time period a new group has been constructed and will be identical at all correct processors, e.g., $g_3 = \{p_1, p_2\}$ (cf. figure 8.1).

In the period between the end of a configuration change and the beginning of the next change the members of the corresponding group do not change, i.e., the view on group membership will be the same during this time period at all processors in the network. Thus, a request for group membership information by a client process at some correct processor would lead to a notification by the corresponding server process. Obviously, this is not true for the time period between the initiation of a new group membership and its end, i.e., during the periods ε_s and ε_f. A client's request would now lead to a situation that is more complex and leaves more choices for reaction. For instance, a server might reject any request during this unstable period or a server takes the old view as long as no new view is constructed. ◇

To implement a group membership service a group membership protocol is needed to provide the client processes at all processors with the information about the actual group membership and to notify to the clients the changes in case of failing processors. A group membership protocol makes use of an underlying communication system. As in [19, 22], we will assume here that the communication system provides the service of an atomic broadcast protocol. Thus, the lowest level, i.e., the communication network will be hidden for a group membership protocol. The whole scenario is schematically illustrated in figure 8.2.

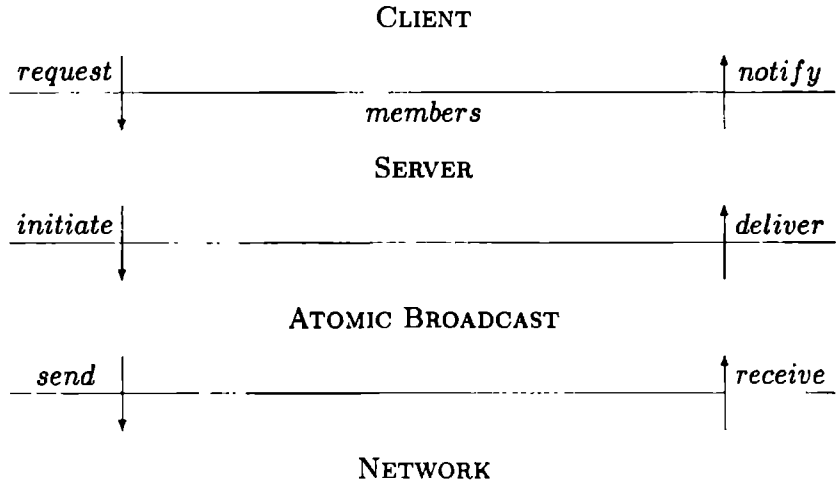


Figure 8.2: Group Membership Service

The two cases as mentioned in the example above, namely that a request *can be notified* because of group stability and that a request *cannot be notified* because of unstability in group membership are distinguished from each other: in case of stability a server makes use of a "short circuit" between *request* and *notify* to provide the requested information. In figure 8.2, this is denoted by *members*. In case of unstability a server makes use of the underlying atomic broadcast protocol to construct a new group membership before again being able to notify clients' requests.

In [19] as well as in [22], however, not the group membership service as given by the "short circuit" *members* (cf. figure 8.2) but that part that is involved to consistently maintain the actual view of group membership is investigated. This part is characterized in [22] informally by the following properties:

- ▷ *Stability of Local Views*: after a processor joins a group it stays joined to that group until a failure is detected or a join occurs.

- ▷ *Agreement on Sequences of Group Identifiers*: During a certain time period, let two correct processors both be joined to a sequence of groups $g_1^1, \ldots, g_n^1$ and $g_1^2, \ldots, g_m^2$, respectively. If the two processors have the same view of the group membership of the first group, i.e., $g_1^1 = g_1^2$ and of the last group, i.e., $g_n^1 = g_m^2$ then the number of groups in both sequences is equal, i.e., $n = m$ and all groups are identical, i.e., $g_i^1 = g_i^2$ for all $1 \leq i \leq n$.

- ▷ *Agreement on Group Membership*: if two correct processors are joined to the same group then they have the same view of the membership of that group.

- ▷ *Reflexivity*: a processor that is joined to a group is a member of that group.

- ▷ *Bounded Join Delays*: there exists a time constant $\varepsilon_s > 0$ such that if a processor starts up at some point in time and stays correct for ε_s time units then all processors that are correct for ε_s time units (including the started one) will join the same group within ε_s time units.

- ▷ *Bounded Failure Detection Delays*: there exists a time constant $\varepsilon_f > 0$ such that if a processor that is joined to some group g, say, has become incorrect at some point in time then all processors that are joined to g and that are correct for ε_f time units will join a group g' to which the incorrect processor will not belong.

The trivial solution of setting the view on group membership to the empty set is ruled out by the reflexivity requirement. The other trivial solution of setting the view to the set of all processors is avoided by the requirements of bounded join and failure detection delays.

8.2 SERVICE SPECIFICATION

As mentioned above we will confine ourselves here to the specification of that part of the processor group membership service that is responsible for maintaining the group membership view.[2] That is, we will *not* be concerned with the properties of the relation between the clients' group membership request and the servers' notification but with the properties of constructing and maintaining the actual view on group membership. Instead of request and notification events we will be concerned with events corresponding, on the one hand, to startup and failure of processors and, on the other hand, to processors' joining and membership of a particular group. A sample of a temporal picture at a certain location is illustrated in figure 8.3.

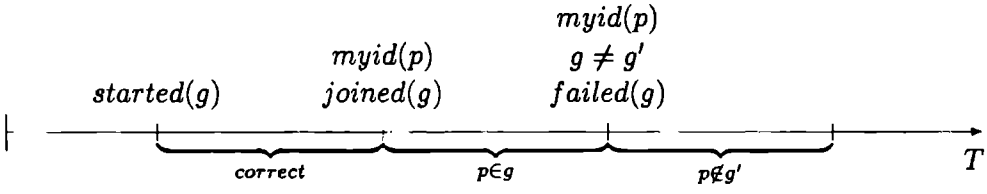


Figure 8.3: A Sample of a Temporal Picture

According to [22] two time constants will be introduced for the specification of liveness properties:[3] in case of a starting processor the *startup delay* is the time between starting to join a group and its completion, i.e., being joined to that group. In case of a failing processor the *failure detection delay* is the time between the failure of a processor and the completion of a join of a group to which the faulty processor does not belong.

The specification of the group membership service will be given in terms of the relation between starting and failing processors, on the one hand, and being joined to a group and processor identifications as elements of a group, on the other hand.[4] The startup of a processor is interpreted here as *starting to join a particular group* and failure of a processor is similarly interpreted as *failing to join a particular group.* A processor that has failed will not be correct until restart.

[2]Note that, in the sequel, we will also refer to this part of a processor group membership service as a group membership service and abbreviate it by *GMS*.

[3]Note that in [19] another choice for time constants has been taken: a *join delay*, a *departure detection delay*, a *group formation delay*, and a *group change delay*.

[4]Note that no distinction will be made between first startup of a processor and its restart after a failure.

For the specification of the group membership service we will need two sets: a set G of group identifiers and set P of processor identifiers[5]. Furthermore, we shall need the following predicates:[6] (let, thereby, g, g' and p be variables ranging over G and P, respectively)

- ▷ *p-correct*: the associated processor is correct
- ▷ *ch-correct*: the associated set of channels is correct
- ▷ *c-correct*: the associated clock is correct
- ▷ $started(g)$: has started to join group g
- ▷ $failed(g)$: has failed to join group g
- ▷ $joined(g)$: is joined to group g
- ▷ $myid(p)$: the processor identifier is p
- ▷ $p \in g$: processor identification p is an element of group g; this construction exactly provides the view on processor group membership that obviously depends on space and time points
- ▷ ε_s: time constant > 0 representing the amount of time between startup of a processor and being joined to a group by all correct processors; it is called *startup delay* of a group membership service[7]
- ▷ ε_f: time constant > 0 representing the amount of time between failure of a processor and being joined to a group to which the faulty processor does not belong or equivalently being deleted from the old group; it is called *failure detection delay* of a group membership service

As in the case of the atomic broadcast service (cf. chapter 7), the subdivision or names of the properties characterizing the group membership service could slightly differ from that originally presented in [22]. This is mainly caused by our special approach of a locative temporal logic.

[5]The need for processor identifiers at this place justifies, in restrospect on chapter 3, our decision to distinguish between meta-level space notion and object-level space notion.

[6]Note that the predicates are indefinite w.r.t. space and time (cf. section 4.1).

[7]In [22], this time constant is called *join delay*. We have changed the name because in both cases starting and failure a join handling must be present so that joining alone is not characteristic for the one or the other case.

ABBREVIATIONS For notational convenience, let us first introduce some abbreviations concerning correctness of local hardware resources. Recall from chapter 5 that local hardware correctness means correctness of the associated processor, clock, and channels:

(1) $$correct \stackrel{\text{def}}{=} p\text{-}correct \wedge ch\text{-}correct \wedge c\text{-}correct$$

Correctness of a starting processor means correctness for at least ε_s time units in the future:

(2) $$correct(\varepsilon_s) \stackrel{\text{def}}{=} \boxplus^{\mathrm{T}}_{\leq \varepsilon_s}\ correct$$

Correctness during failure detection handling means correctness of the corresponding processor for ε_f time units:

(3) $$correct(\varepsilon_f) \stackrel{\text{def}}{=} \boxplus^{\mathrm{T}}_{\leq \varepsilon_f}\ correct$$

Moreover, we need the following abbreviation indicating that a processor identifier is not contained in a group:

(4) $$p \notin g \stackrel{\text{def}}{=} \neg(p \in g)$$

STARTUP LIVENESS When a processor has started up to join a group and is correct for at least ε_s time units then all correct processors (including the started one) must be joined to that group by ε_s time units:

(5) $$started(g) \wedge correct(\varepsilon_s)$$
$$\rightarrow$$
$$\boxdot^{\mathrm{L}}[correct(\varepsilon_s) \rightarrow \diamondplus^{\mathrm{T}}_{\leq \varepsilon_s}(correct \wedge joined(g))]$$

This is exactly the old property of bounded join delays.

FAILURE LIVENESS When a processor has failed to join a group then there exists a group to which all correct processors must be joined by ε_f time units:

(6) $$failed(g) \wedge myid(p)$$
$$\rightarrow$$
$$\exists g' : \boxdot^{\mathrm{L}}\Big[correct(\varepsilon_f) \rightarrow \diamondplus^{\mathrm{T}}_{\leq \varepsilon_f}(correct \wedge joined(g'))\Big]$$

This is the liveness part of the old property of bounded failure detection delays, that is, the property that the identifier of the failed processor will not be contained in the new group will be considered separately in the next property.

FAILURE SAFETY When a processor has joined a particular group and there exists a location where the corresponding processor is not correct since it has failed and the identifier of the latter processor was not contained in that group before its failure then the identifier of the failed processor will not be contained in that group (provided that all local hardware resources are correct):

(7)
$$
\begin{array}{l}
[\, \mathit{correct} \wedge \mathit{joined}(g) \\
\wedge \\
\quad \Diamond^{\mathrm{L}} [\, \neg \mathit{p\text{-}correct} \wedge \mathit{myid}(p) \\
\qquad \wedge \\
\qquad\quad [\, \neg \mathit{p\text{-}correct} \\
\qquad\quad\ \mathrm{S} \\
\qquad\quad\ (\mathit{failed}(g') \wedge \boxminus^{\mathrm{T}} (p \notin g \wedge \mathit{myid}(p))) \\
\qquad\quad] \\
\qquad] \\
] \\
\rightarrow \\
p \notin g
\end{array}
$$

LOCAL GROUP MEMBERSHIP When a processor is joined to a group then its identifier must be contained in that group (provided that all local hardware resources are correct):

(8)
$$\mathit{correct} \wedge \mathit{joined}(g) \wedge \mathit{myid}(p) \rightarrow p \in g$$

This property provides the old reflexivity property as mentioned above.

LOCAL STABILITY When a correct processor is joined to a particular group then it stays to be correct and joined to that group until a failure occurs or it is joined to another group (provided that all local hardware resources are correct):

(9)
$$
\begin{array}{lll}
\mathit{correct} \wedge \mathit{joined}(g) & \rightarrow & (\mathit{correct} \wedge \mathit{joined}(g)) \\
 & & \mathrm{U}\ [\, \mathit{failed}(g) \\
 & & \quad \vee \exists g' : (g \neq g' \wedge \mathit{correct} \wedge \mathit{joined}(g')) \\
 & & \quad]
\end{array}
$$

This property expresses the old property of stability of local views.[8]

[8]The name has changed because it is not a statement on local views but on local joins.

AGREEMENT ON GROUP MEMBERSHIP When a processor is joined to a particular group and its identifier is contained in that group then this view must be the same for every correct processor that is joined to that group (provided that all local hardware resources are correct):

$$(10) \quad correct \wedge joined(g) \wedge p \in g \rightarrow \neg\nabla^{L}(correct \wedge joined(g) \wedge p \notin g))$$

This is exactly the old property of agreement on group membership.

AGREEMENT ON GROUP ORDERING Agreement on group ordering is reached when all correct processors that are joined to two groups in a particular order are joined to these groups in the same order:

$$(11) \quad [\ (correct \wedge joined(g)) \wedge \diamondsuit^{T}(correct \wedge joined(g'))\ ] \rightarrow \boxdot^{L}[\ \neg\boxbslash^{T}[\ (correct \wedge joined(g')) \wedge \diamondsuit^{T}(correct \wedge joined(g))\ ]\ ]$$

Note that this property differs from the old property of agreement on sequences of group identifiers in the sense that we regard pairs of group identifiers and not a sequence. Our property is stronger and thus implies the other one.

UNIQUENESS The group to which a correct processor is joined must be unique:

$$(12) \quad correct \wedge joined(g) \wedge joined(g') \rightarrow g = g'$$

The group that a processor has started or failed to join, respectively, must be unique in space, in time, in space and time, and w.r.t. the group identifier:

$$(13) \quad started(g) \wedge started(g') \rightarrow g = g'$$

$$(14) \quad started(g) \rightarrow \neg\nabla^{T} started(g)$$

$$(15) \quad started(g) \rightarrow \neg\nabla^{LT} started(g)$$

$$(16) \quad failed(g) \wedge failed(g') \rightarrow g = g'$$

$$(17) \quad failed(g) \rightarrow \neg\nabla^{T} failed(g)$$

$$(18) \quad failed(g) \rightarrow \neg\nabla^{LT} failed(g)$$

Note that it will not be excluded that at the same point in time another processor started or failed to join the same group, respectively. All equations containing predicates *started* or *failed* provide environment assumptions which must not be proved afterwards.

In [22], uniqueness properties are only considered for processor identifiers (cf. section 8.3.2 below). All other uniqueness properties as mentioned above are due to our special approach (see also chapter 7).

8.3 A CLASS OF PROTOCOLS

In the introduction, we took the decision not to provide any programming language and to confine ourselves in this thesis to the specification of distributed real-time systems. Therefore, we shall only give a specification characterizing not exactly one particular protocol but a class of *group membership protocols (GMP)*. Afterwards, we will formally prove that this class satisfies the properties of the group membership service.

Amongst the possible implementations of the group membership service is a so called *periodic broadcast membership protocol* [9] as mentioned in [19, 22]. This protocol provides a simple mechanism for detecting processor startups and failures. It is less efficient in the case that only a few or even no failures occur because messages will be sent periodically by each correct processor to indicate its presence to all other processors in the system. According to [19, 22], a periodic broadcast membership protocol makes use of an atomic broadcast service (cf. chapter 7). An illustration is given in figure 8.4.

Recall that we have confined ourselves previously to that part of the group membership problem that is concerned with the maintenance of the actual view on processor group membership. The specification of the periodic broadcast membership protocol will be given here in the following terms: in case of processor startup a message of type "new" will be sent to all correct processors in the system to indicate that a new group membership has to be created. Each correct processor has to respond to such a message with a message of type "present" to indicate its correctness and presence in the network. In case of processor failure the corresponding processor cannot do anything. Therefore, it will be required that each correct processor indicates from time to time that it is still correct and present in the network. This is done by sending periodically messages of type "present" by all correct processors.

In addition to the set G of group identifiers, the set P of processor identifiers, and the predicates and abbreviations as introduced in section 8.2, we will make use of some new notions in the specification formulae of the periodic broadcast membership protocol. So, we presuppose a set $M = \{\mathrm{Ne}, \mathrm{Pr}\}$ of only two messages one denoting a message of type "new" and the other

[9] We shall also refer to the periodic broadcast membership protocol by the abbreviation *GMP*.

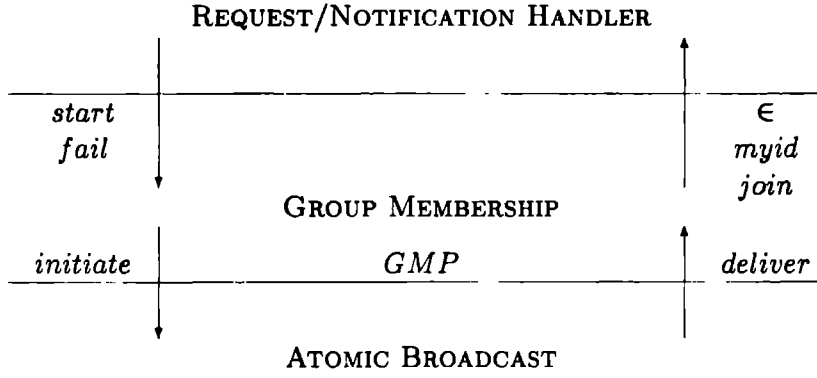


Figure 8.4: Group Membership Protocols

denoting a message of type "present". Furthermore, let g, g', p, p', and m, m' be variables ranging over G, P, and M, respectively:[10]

- ▷ *initiated*(m, g, p): broadcast of an m-type message for group identifier g and processor identifier p has been initiated
- ▷ *delivered*(m, g, p): an m-type message for group identifier g and processor identifier p has been delivered
- ▷ μ: time constant > 0 representing the amount of time between two consecutive broadcast initiations with Pr-type messages; it is called *check time period* of a periodic broadcast membership protocol
- ▷ ε: *diffusion time* constant > 0 of the atomic broadcast service

8.3.1 FAULT-HYPOTHESES

Distributed real-time systems are, in general, sensitive to failures. Therefore, we have to characterize those types of faults that may occur without any impacts on the systems correct behaviour. This is usually done by stating the *fault-hypothesis (FH)* of the system or parts of it. According to our view of

[10]Note that the predicates are indefinite w.r.t. space and time (cf. section 4.1).

processes and distributed real-time systems (cf. chapter 5), fault-hypotheses will be provided for clocks, channels, processors, and the whole network.

ABBREVIATION For notational convenience, let us introduce an abbreviation for the correctness of all network resources associated with a location:

(19) $$p\text{-}ch\text{-}correct \stackrel{\text{def}}{=} p\text{-}correct \wedge ch\text{-}correct$$

FH CLOCKS All clocks are assmuned to be correct independent of the actual locative temporal reference point:

(20) $$c\text{-}correct$$

Although we know that clock synchronization is an essential problem in distributed real-time systems we leave it out here (see chapter 9) and presuppose a central clock system. We could also think, e.g., of a distributed clock system but then we would have to presuppose exactly synchronized clocks. Note that this differs from [20] in that there *approximately* synchronized clocks have been assumed.

FH CHANNELS All channels are assumed to be correct independent of the actual locative temporal reference point:

(21) $$ch\text{-}correct$$

FH PROCESSORS When a processor has failed to join a particular group then that processor will not be correct and it will remain to be incorrect for at least ε_f time units:

(22) $$failed(g) \rightarrow \boxplus^{\mathrm{T}}_{\leq \varepsilon_f} \neg p\text{-}correct$$

A processor that is incorrect will neither initiate nor deliver any message (*fail-silence* property):

(23) $$\neg p\text{-}correct \rightarrow (\neg initiated(m, g, p) \wedge \neg delivered(m, g, p))$$

Observe that opposed to chapter 7 a processor here is not assumed to be fail-stop. This is because group membership must take into account not only failure but also startup of processors.

FH NETWORK PARTITIONING For all pairs of locations with a distance > 1 where the corresponding processor and set of channels are correct there

must exist at least one intermediate location where the corresponding processor and set of channels are also correct:

$$(24) \qquad \forall \zeta : \; [\quad 1 < \zeta \wedge \textit{p-ch-correct} \wedge \diamond^{L}_{=\zeta} \textit{p-ch-correct} \;\rightarrow\; \exists \zeta' : 0 < \zeta' < \zeta \wedge \diamond^{L}_{=\zeta'} \textit{p-ch-correct} \;]$$

This formula characterizes the assumption of no partitions in a dynamically evolving physical network, as described in [22]. It follows that the network is at least 1-valent at all correct processors (cf. chapter 3).

8.3.2 PROTOCOL SPECIFICATION

There are mainly three tasks that have to be considered for the periodic broadcast membership protocol: *startup handling*, *failure detection handling*, and *group validation*. Additionally we have to regard once more the *atomic broadcast service* and the relation between the primitive notions of the group membership service and the primitive notions of this class of group membership protocols.

Startup and failure detection mainly differ in that startup can be indicated to all others in the network by the corresponding processor itself. In contrary, failure of a processor must be detected by others in the network because one cannot expect from a "dead" processor that it indicates itself the death. As a consequence the time periods and groups involved in the corresponding actions will differ.

Processor startup leads to a temporal picture as presented in figure 8.5. The first period is the time for broadcasting a message of type "new" to all correctly functioning processors in the network to invite them to form a new group.[11] The second period is the time for broadcasting response messages of type "present" to indicate that the corresponding processor is still functioning correctly.

Both time periods are presupposed here to be always equal to the diffusion time of the underlying atomic broadcast service. No time difference is assumed between delivery of the Ne-type message and the initiation of the broadcast of the corresponding Pr-type message.

Processor failure detection leads to a temporal picture as provided in figure 8.6. A processor will regard itself to be invited to join a (new) group

[11] The time point characterized here by the predicate *delivered*(Ne, g, p) is called *view time* in [22].

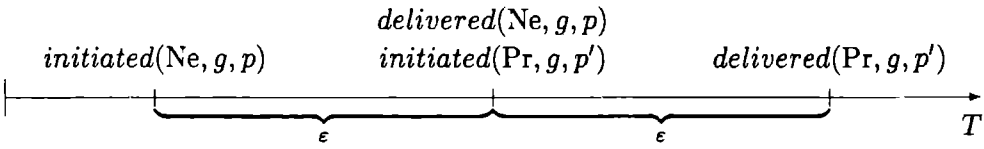


Figure 8.5: Phases in Case of Startup Indication

periodically, that is, every μ time units after the last invitation each processor automatically initiates a broadcast for a Pr-type message to indicate to all others in the network that it is itself still functioning correctly.[12] The second time period is assumed here to be always equal to the diffusion time of the underlying atomic broadcast service.[13]

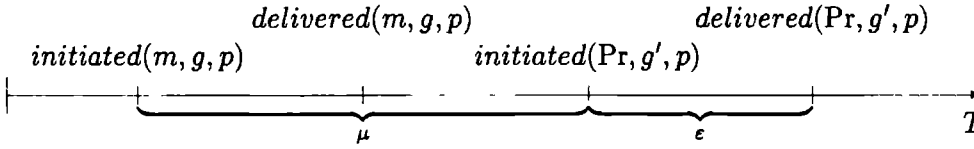


Figure 8.6: Phases in Case of Failure Detection

A complication could occur when between initiating a group validation and its completion a second group validation is initiated, e.g., in case of a processor startup during validation. Then, two invitations to join different groups would be open. In [20, 22], this problem is resolved by requiring that all initiated but not already completed broadcasts except the last one will be cancelled. Opposed to that we will exclude any initiations of group validations during this critical instable time period and require instead atomicity of group validation. This decision is not quite realistic but it simplifies our reasoning and leaves our main purposes here untouched.

ABBREVIATIONS For notational convenience, let us introduce abbreviations concerning correctness of local hardware resources between the initiation of two successive broadcasts:

$$correct(\mu) \stackrel{\text{def}}{=} \circledast_{\leq \mu}^{T} correct \tag{25}$$

[12]The time point characterized here by the predicate *initiated*(Pr, g', p) is called *check time* in [22].

[13]The time point characterized here by the predicate *delivered*(Pr, g', p) is called *confirmation time* in [22].

during a broadcast itself:

(26) $$correct(\varepsilon) \stackrel{\text{def}}{=} \boxplus^{\mathrm{T}}_{\leq \varepsilon}\ correct$$

between two successive broadcasts and during the second broadcast:

(27) $$correct(\mu \oplus \varepsilon) \stackrel{\text{def}}{=} \boxplus^{\mathrm{T}}_{\leq (\mu \oplus \varepsilon)}\ correct$$

during two successive broadcasts:

(28) $$correct(\varepsilon \oplus \varepsilon) \stackrel{\text{def}}{=} \boxplus^{\mathrm{T}}_{\leq (\varepsilon \oplus \varepsilon)}\ correct$$

BROADCAST ON INVITATION When a processor has been invited to join a particular group then it initiates a broadcast of a Pr-type message with its own processor identifier to indicate that it is itself correct (provided that all local hardware resources are correct):

(29) $$correct \wedge delivered(\mathrm{Ne}, g, p) \rightarrow \exists p' : (myid(p') \wedge initiated(\mathrm{Pr}, g, p'))$$

Observe that there is no difference between the time point where a Ne-type message has been delivered and the time point where the broadcast of a Pr-type message will have been initiated as response.

LOCAL MEMBERSHIP When a correct processor has delivered a Pr-type message with its own identifier for a particular group then the identifier will be contained in that group (provided all local hardware resources are correct):

(30) $$correct \wedge delivered(\mathrm{Pr}, g, p) \wedge myid(p) \rightarrow p \in g$$

MEMBERSHIP When a correct processor has delivered a Pr-type message for a particular group and processor identifier then the identifier will be contained in that group (provided all local hardware resources are correct):

(31) $$correct \wedge delivered(\mathrm{Pr}, g, p) \rightarrow p \in g$$

MEMBERSHIP SAFETY When a processor identifier is contained in a group then a Pr-type message for that group and processor identifier has been delivered (provided all local hardware resources are correct):

(32) $$correct \wedge p \in g \rightarrow \diamondsuit^{\mathrm{T}}\ (correct \wedge delivered(\mathrm{Pr}, g, p))$$

When a processor identifier is not contained in a group then a Pr-type message for that group and processor identifier must not have been delivered (provided all local hardware resources are correct):

(33) $$correct \wedge p \notin g \rightarrow \neg \Diamond^{\mathrm{T}} (correct \wedge delivered(\mathrm{Pr}, g, p))$$

When the identifier of the associated processor is not contained in a group then a broadcast of a Pr-type message for that group and processor identifier cannot have been initiated:

(34) $$p \notin g \wedge myid(p) \rightarrow \boxdot^{\mathrm{T}} (\neg initiated(\mathrm{Pr}, g, p))$$

MEMBERSHIP VIEW UNIQUENESS A correct processor will not have different views (provided all local hardware resources are correct):

(35) $$correct \wedge p \in g \wedge p \in g' \rightarrow g = g'$$

STABILITY OF MEMBERSHIP When a processor identifier is contained in a particular group then it remains in that group until a failure occurs or a message with a new group identifier will have been delivered (provided that all local hardware resources are correct):

(36)
$$\begin{array}{lll} (correct \wedge p \in g) & \rightarrow & (correct \wedge p \in g) \\ & & \mathrm{U}\ [\ failed(g) \\ & & \quad \vee \\ & & \quad \exists m, g', p' : (\ correct \\ & & \qquad\qquad \wedge\ g \neq g' \\ & & \qquad\qquad \wedge\ delivered(m, g', p') \\ & & \qquad\qquad \wedge\ myid(p') \\ & & \qquad\qquad)\ \\ & &] \end{array}$$

ATOMICITY OF VALIDATION During validation of a group, i.e., between delivery of a message and the initiation of the corresponding broadcast any further initiation will be disallowed (provided that all local hardware resources are correct):

(37) $$(correct \wedge delivered(m, g, p)) \rightarrow$$

$$\neg \Diamond^{\mathrm{T}} [\ (correct \wedge initiated(m', g', p)) \wedge \Diamond^{\mathrm{T}}(correct \wedge initiated(m, g, p))\ ]$$

In the sequel, we will recall from chapter 7 the properties of an atomic broadcast service (ABS), i.e., atomic liveness, safety, and ordering. Of course, the formulae must be put in a suitable form for the purposes in this chapter but this has not a great impact.

ABS ATOMIC LIVENESS If the diffusion of a message for a particular group and the own identifier has been initiated then that message will have been deliveredwithin ε time units at all locations (provided that all local hardware resources are correct):

(38) $$correct \wedge initiated(m, g, p) \wedge myid(p) \rightarrow \boxdot^{\mathrm{L}}[correct(\varepsilon) \rightarrow \Diamond^{\mathrm{T}}_{\leq \varepsilon}(correct \wedge delivered(m, g, p))]$$

ABS SAFETY A message that has been delivered must have been initiated somewhere sometime where the processor identifier corresponds to that processor (provided that all local hardware resources are correct):

(39) $$correct \wedge delivered(m, g, p) \rightarrow \Diamond^{\mathrm{LT}} [correct \wedge initiated(m, g, p) \wedge myid(p)]$$

ABS ORDERING Two messages that are delivered in a particular order for a particular group and the corresponding processor identifier must have been delivered for that group in the same order everywhere (provided that all local hardware resources are correct):

(40) $$[\ (correct \wedge delivered(m, g, p) \wedge myid(p)) \wedge \Diamond^{\mathrm{T}} (correct \wedge delivered(m', g', p) \wedge myid(p))\ ] \rightarrow \boxdot^{\mathrm{L}}[\ \neg \nabla^{\mathrm{T}}\ [\ (correct \wedge delivered(m', g', p') \wedge myid(p')) \wedge \Diamond^{\mathrm{T}} (correct \wedge delivered(m, g, p') \wedge myid(p))\ ]\ ]$$

Before being able to prove that the protocol specification is a specification refinement of the service specification we have to relate the corresponding primitive notions, that is, startup and failure of processors on the higher level must be related to initializations of message broadcasts on the lower level and deliveries of messages on the lower level must be related to group joining and group membership on the higher level.

PROCESSOR IDENTIFIERS Processor identifiers exist for all processors whether correct or not correct:

(41) $$\exists p : myid(p)$$

Processor identifiers will be unique in space, in time, and w.r.t. identifier itself. A processor will not have two different identifiers:

(42) $$myid(p) \wedge myid(p') \rightarrow p = p'$$

Different processors will have different identifiers:

(43) $$myid(p) \rightarrow \neg\nabla^{\mathrm{L}} myid(p)$$

The identifier of a processor will always be the same:

(44) $$myid(p) \rightarrow \neg\nabla^{\mathrm{T}}(\neg myid(p))$$

START IMPLIES INITIATION When a correct processor has started to join a group then it will have been initiated a broadcast of a Ne-type message containing its own processor identifier:

(45) $$correct \wedge started(g) \wedge myid(p) \rightarrow initiated(\mathrm{Ne}, g, p)$$

INITIATION IMPLIES START When a correct processor has initiated the broadcast of a Ne-type message with its own processor identifier and for a particular group then the corresponding processor must have started up to join that group:

(46) $$correct \wedge initiated(\mathrm{Ne}, g, p) \wedge myid(p) \rightarrow started(g)$$

FAILURE IMPLIES INITIATION When a processor has failed to join a particular group then there will be an initiation of a broadcast of a Pr-type message

for a different group and the corresponding processor identifers within μ time units:

$$
\begin{aligned}
(47) \qquad & \mathit{failed}(g) \wedge \mathit{myid}(p) \\
& \rightarrow \\
& \exists g' : \boxdot^{\mathrm{L}}[\ \mathit{correct}(\mu) \\
& \qquad \rightarrow \\
& \qquad \diamondsuit^{\mathrm{T}}_{\leq \mu}\ \exists p' : (\ \mathit{correct} \\
& \qquad\qquad \wedge\ \mathit{initiated}(\mathrm{Pr}, g', p') \\
& \qquad\qquad \wedge\ \mathit{myid}(p') \\
& \qquad\qquad \wedge\ g \neq g' \\
& \qquad\qquad)\ \\
&]
\end{aligned}
$$

This characterizes the liveness of an periodic broadcast protocol in case of failing processors.

DELIVERY IMPLIES JOINING When a correct processor has delivered a Pr-type message for a particular group with its own identifier then it is joined to that group:

$$(48) \qquad \mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p) \wedge \mathit{myid}(p) \rightarrow \mathit{joined}(g)$$

JOINING IMPLIES DELIVERY When a correct processor is joined to a particular group then a Pr-type message must have been delivered (now or previously) for that group and the own processor identifier:

$$
\begin{aligned}
(49) \qquad & \mathit{correct} \wedge \mathit{joined}(g) \wedge \mathit{myid}(p) \\
& \rightarrow \\
& \diamondsuit^{\mathrm{T}}\ (\mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p))
\end{aligned}
$$

MEMBERSHIP IMPLIES JOINING When the identifier of a correct processor is contained in a particular group then it is also joined to that group:

$$(50) \qquad \mathit{correct} \wedge p \in g \wedge \mathit{myid}(p) \rightarrow \mathit{joined}(g)$$

JOINING IMPLIES MEMBERSHIP When a correct processor is joined to a particular group then its own identifier must be contained in that group:

$$(51) \qquad \mathit{correct} \wedge \mathit{joined}(g) \wedge \mathit{myid}(p) \rightarrow p \in g$$

In table 8.1, we have set in relation the most important properties of the group membership service and the periodic broadcast membership protocol as defined above.

GMS	*GMP*
– group identifiers	– group identifiers
– processor identifiers	– processor identifiers
	– messages
– startup liveness	– broadcast on invitation, ABS atomic liveness
– failure liveness	– periodic broadcast ABS atomic liveness
– failure safety	– ABS safety
– local group membership	– membership safety, ABS safety
– local stability	– stability of membership
– agreement on group membership	– membership view uniqueness
– agreement on group ordering	– ABS ordering
– join uniqueness	– membership view uniqueness
	– fault-hypotheses

Table 8.1: Relation between *GMS* and *GMP*

Observe that processor group membership as discussed above is also a good example of how data domains on different abstraction levels are related to one another, e.g., group, processor, and message domains on the lower level and group and processor domains on the higher level.

8.3.3 Correctness

We now proceed to prove that the class of periodic broadcast membership protocols as defined by the specification formulae above is correct w.r.t. the specification of a group membership service as mentioned previously. That is, we will prove that the properties of *startup liveness*, *failure liveness*, *failure safety*, *local group membership*, *local stability*, *agreement on group membership*,

agreement on group ordering, and *join uniqueness* can be derived from the properties of GMP provided that the clocks, channels, and processors meet their fault-hypotheses, respectively. This is stated in the following

THEOREM 8.1 Let GMP be a distributed service implementing the periodic broadcast membership protocol and let Σ_{GMP} be its specification as defined above. Furthermore, let $0 < \mu$ be the check time period of the GMP, let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service, and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Then, GMP implements the group membership service, i.e.,[14]

$$\begin{aligned}\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ & \text{STARTUP LIVENESS} \\ & \wedge \text{FAILURE LIVENESS} \\ & \wedge \text{FAILURE SAFETY} \\ & \wedge \text{LOCAL GROUP MEMBERSHIP} \\ & \wedge \text{LOCAL STABILITY} \\ & \wedge \text{AGREEMENT ON GROUP MEMBERSHIP} \\ & \wedge \text{AGREEMENT ON GROUP ORDERING} \\ & \wedge \text{JOIN UNIQUENESS} \\ &]\end{aligned}$$

$\Diamond$

Because of the nature of the theorem, that is, an implication where the succedent is a conjunction the proof will proceed in such a way that we first prove each conjunct separately and then gather the corresponding results so that the proof of the theorem becomes itself simple.

LEMMA 8.1 (STARTUP LIVENESS) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying

[14]The names STARTUP LIVENESS, FAILURE LIVENESS, FAILURE SAFETY, LOCAL GROUP MEMBERSHIP, LOCAL STABILITY, AGREEMENT ON GROUP MEMBERSHIP, AGREEMENT ON GROUP ORDERING, and JOIN UNIQUENESS denote the equations 5, 6, 7, 8, 9, 10, 11, and 12, respectively.

atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ started(g) \wedge correct(\varepsilon_s) \rightarrow \boxdot^{\text{L}} [\ correct(\varepsilon_s) \rightarrow \diamondplus^{\text{T}}_{\leq \varepsilon_s}(correct \wedge joined(g))\ ]\ ]$$

◇

Proof 8.1 (Lemma 8.1)

1. $\varepsilon > 0$ — Assumption
2. $\mu > 0$ — Assumption
3. $started(g) \wedge correct(\varepsilon_s)$ — Assumption
4. $started(g) \wedge \boxplus^{\text{T}}_{\leq \varepsilon_s} correct$ — 3., (2)
5. $started(g) \wedge correct$ — 4., (LTL-53), (LTL-52), (LTL-50)
6. $started(g) \wedge correct \wedge \exists p : myid(p)$ — 5., (41), (LTL-80), (LTL-136)
7. $\exists p : (myid(p) \wedge correct \wedge initiated(\text{Ne}, g, p))$ — 6., (45), (LTL-80), (LTL-136)
8. $\exists p : \boxdot^{\text{L}}[correct(\varepsilon) \rightarrow \diamondplus^{\text{T}}_{\leq \varepsilon}\ (correct \wedge delivered(\text{Ne}, g, p))]$ — 7., (38), (LTL-136)
9. $\exists p :$ $\boxdot^{\text{L}}[correct(\varepsilon) \rightarrow \diamondplus^{\text{T}}_{\leq \varepsilon} \exists p' : (correct \wedge myid(p') \wedge initiated(\text{Pr}, g, p'))]$ — 8., (29), (LTL-80), (LTL-136)
10. $\boxdot^{\text{L}}[correct(\varepsilon) \rightarrow \diamondplus^{\text{T}}_{\leq \varepsilon} \exists p' : (correct \wedge myid(p') \wedge initiated(\text{Pr}, g, p'))]$ — 9., (LTL-83), (LTL-136)
11. $\boxdot^{\text{L}}[\ correct(\varepsilon) \rightarrow \diamondplus^{\text{T}}_{\leq \varepsilon} \exists p' : [\ myid(p')$

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad \wedge \\
\qquad\qquad\qquad\qquad\quad \boxdot^{\mathrm{L}}(\mathit{correct}(\varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (\mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p'))) \\
\qquad\qquad\qquad\qquad] \\
\qquad] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad 10., (38), (\text{LTL-80}), (\text{LTL-136})
\end{array}
$$

12. $\boxdot^{\mathrm{L}}[\ \mathit{correct}(\varepsilon)$

$$
\begin{array}{l}
\qquad \rightarrow \\
\qquad\quad \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \exists p' : [\ \mathit{myid}(p') \\
\qquad\qquad\qquad\qquad \wedge \\
\qquad\qquad\qquad\qquad\quad (\mathit{correct}(\varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (\mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p'))) \\
\qquad\qquad\qquad\qquad] \\
\qquad] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad 11., (\text{LTL-86}), (\text{LTL-136})
\end{array}
$$

13. $\boxdot^{\mathrm{L}}[\ \mathit{correct}(\varepsilon)$

$$
\begin{array}{l}
\qquad \rightarrow \\
\qquad\quad \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \exists p' : [\ \mathit{correct}(\varepsilon) \\
\qquad\qquad\qquad\qquad \rightarrow \\
\qquad\qquad\qquad\qquad\quad (\mathit{myid}(p') \wedge \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (\mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p'))) \\
\qquad\qquad\qquad\qquad] \\
\qquad] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad 12., (\text{LTL-80}), (\text{LTL-136})
\end{array}
$$

14. $\boxdot^{\mathrm{L}}[\ \mathit{correct}(\varepsilon)$

$$
\begin{array}{l}
\qquad \rightarrow \\
\qquad\quad \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \exists p' : [\ \mathit{correct}(\varepsilon) \\
\qquad\qquad\qquad\qquad \rightarrow \\
\qquad\qquad\qquad\qquad\quad \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (\mathit{myid}(p') \wedge \mathit{correct} \wedge \mathit{delivered}(\mathrm{Pr}, g, p')) \\
\qquad\qquad\qquad\qquad] \\
\qquad] \qquad 13., (44), \text{PROPOSITION A.15}, \text{PROPOSITION A.16}, (\text{LTL-136})
\end{array}
$$

15. $\boxdot^{\mathrm{L}}[\ \mathit{correct}(\varepsilon)$

$$
\begin{array}{l}
\qquad \rightarrow \\
\qquad\quad \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \exists p' : [\mathit{correct}(\varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (\mathit{correct} \wedge \mathit{joined}(g))] \\
\qquad] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad 14., (48), (\text{LTL-80}), (\text{LTL-136})
\end{array}
$$

16. $\boxdot^{\mathrm{L}}[\ correct(\varepsilon)$

$\rightarrow$

$\diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ [\ correct(\varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (correct \wedge joined(g))]$

$]$ 15., (41), (LTL-136)

17. $\boxdot^{\mathrm{L}}[\ correct(\varepsilon)$

$\rightarrow$

$[\ \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ correct(\varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (correct \wedge joined(g))\ ]$

$]$ 16., PROPOSITION A.13, (LTL-136)

18. $\boxdot^{\mathrm{L}}[\ (\boxplus^{\mathrm{T}}_{\leq\varepsilon}\ correct \wedge \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \boxplus^{\mathrm{T}}_{\leq\varepsilon}\ correct)$

$\rightarrow$

$\diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (correct \wedge joined(g))$

$]$ 17., (26), (LTL-80), (LTL-136)

19. $\boxdot^{\mathrm{L}}\big[\boxplus^{\mathrm{T}}_{\leq(\varepsilon\oplus\varepsilon)}\ correct \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (correct \wedge joined(g))\big]$

18., PROPOSITION A.23, (LTL-136)

20. $\boxdot^{\mathrm{L}}[correct(\varepsilon \oplus \varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ \diamondplus^{\mathrm{T}}_{\leq\varepsilon}\ (correct \wedge joined(g))]$

19., (28)

21. $\boxdot^{\mathrm{L}}[correct(\varepsilon \oplus \varepsilon) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon\oplus\varepsilon}\ (correct \wedge joined(g))]$

20., PROPOSITION A.27, (LTL-136)

22. $\boxdot^{\mathrm{L}}[correct(\varepsilon_s) \rightarrow \diamondplus^{\mathrm{T}}_{\leq\varepsilon_s}\ (correct \wedge joined(g))]$

21., (2), PROPOSITION A.26, (LTL-136)

provided that $\varepsilon \oplus \varepsilon \leq \varepsilon_s$

q.e.d.

LEMMA 8.2 (FAILURE LIVENESS) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check

time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ failed(g) \wedge myid(p) \rightarrow \exists g' : \boxdot^{\text{L}} [\ correct(\varepsilon_f) \rightarrow \diamondsuit^{\text{T}}_{\leq \varepsilon_f} (\ correct \wedge joined(g')\ )\ ]\ ]$$

$\Diamond$

PROOF 8.2 (LEMMA 8.2)

1. $\mu > 0$ — Assumption
2. $\varepsilon > 0$ — Assumption
3. $failed(g) \wedge myid(p)$ — Assumption
4. $\exists g' : \boxdot^{\text{L}}[\ (correct(\mu) \rightarrow \diamondsuit^{\text{T}}_{\leq \mu}\ \exists p' : (correct \wedge initiated(\text{Pr}, g', p') \wedge myid(p') \wedge g \neq g')\ ]$ — 3., (47), (LTL-136)
5. $\exists g' : \boxdot^{\text{L}}[\ correct(\mu) \rightarrow \diamondsuit^{\text{T}}_{\leq \mu}\ \exists p' : [\ myid(p') \wedge \boxdot^{\text{L}}(correct(\varepsilon) \rightarrow \diamondsuit^{\text{T}}_{\leq \varepsilon}\ (correct \wedge delivered(\text{Pr}, g, p')))\ ]\ ]$ — 5., (38), (LTL-80), (LTL-136)

6. $\exists g' :$

$$\boxdot^{\mathrm{L}}[\, correct(\mu)$$
$$\rightarrow$$
$$\oplus^{\mathrm{T}}_{\leq \mu}\ \exists p' : [\, myid(p')$$
$$\wedge$$
$$(correct(\varepsilon) \rightarrow \oplus^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge delivered(\mathrm{Pr}, g, p')))$$
$$]$$
$$] \qquad \text{5., (LTL-86), (LTL-136)}$$

7. $\exists g' :$

$$\boxdot^{\mathrm{L}}[\, correct(\mu)$$
$$\rightarrow$$
$$\oplus^{\mathrm{T}}_{\leq \mu}\ \exists p' : [\, correct(\varepsilon)$$
$$\rightarrow$$
$$(myid(p') \wedge \oplus^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge delivered(\mathrm{Pr}, g, p')))$$
$$]$$
$$] \qquad \text{6., (LTL-80), (LTL-136)}$$

8. $\exists g' :$

$$\boxdot^{\mathrm{L}}[\, correct(\mu)$$
$$\rightarrow$$
$$\oplus^{\mathrm{T}}_{\leq \mu}\ \exists p' : [\, correct(\varepsilon)$$
$$\rightarrow$$
$$\oplus^{\mathrm{T}}_{\leq \varepsilon}\ (myid(p') \wedge correct \wedge delivered(\mathrm{Pr}, g, p'))$$
$$]$$
$$] \qquad \text{7., (44), PROPOSITION A.15, PROPOSITION A.16, (LTL-136)}$$

9. $\exists g' :$

$$\boxdot^{\mathrm{L}}[\, correct(\mu)$$
$$\rightarrow$$
$$\oplus^{\mathrm{T}}_{\leq \mu}\ \exists p' : [correct(\varepsilon) \rightarrow \oplus^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))]$$
$$] \qquad \text{8., (48), (LTL-80), (LTL-136)}$$

10. $\exists g' :$
$\boxdot^{\mathrm{L}}[\ correct(\mu)$
$\rightarrow$
$\diamondsuit^{\mathrm{T}}_{\leq \mu}\ [\ correct(\varepsilon) \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))]$
$]$ 9., (41), (LTL-136)

11. $\exists g' :$
$\boxdot^{\mathrm{L}}[\ correct(\mu)$
$\rightarrow$
$[\ \diamondsuit^{\mathrm{T}}_{\leq \mu}\ correct(\varepsilon) \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \mu}\ \diamondsuit^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))\ ]$
$]$ 10., PROPOSITION A.13, (LTL-136)

12. $\exists g' :$
$\boxdot^{\mathrm{L}}[\ (\boxplus^{\mathrm{T}}_{\leq \mu}\ correct \wedge \diamondsuit^{\mathrm{T}}_{\leq \mu}\ \boxplus^{\mathrm{T}}_{\leq \varepsilon}\ correct)$
$\rightarrow$
$\diamondsuit^{\mathrm{T}}_{\leq \mu}\ \diamondsuit^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))$
$]$ 11., (26), (LTL-80), (LTL-136)

13. $\exists g' : \boxdot^{\mathrm{L}}\Big[\boxplus^{\mathrm{T}}_{\leq (\mu \oplus \varepsilon)}\ correct \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \mu}\ \diamondsuit^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))\Big]$
12., PROPOSITION A.23, (LTL-136)

14. $\exists g' : \boxdot^{\mathrm{L}}[correct(\mu \oplus \varepsilon) \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \mu}\ \diamondsuit^{\mathrm{T}}_{\leq \varepsilon}\ (correct \wedge joined(g))]$
13., (28)

15. $\exists g' : \boxdot^{\mathrm{L}}[correct(\mu \oplus \varepsilon) \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \mu \oplus \varepsilon}\ (correct \wedge joined(g))]$
14., PROPOSITION A.27, (LTL-136)

16. $\exists g' : \boxdot^{\mathrm{L}}\Big[correct(\varepsilon_f) \rightarrow \diamondsuit^{\mathrm{T}}_{\leq \varepsilon_f}\ (correct \wedge joined(g))\Big]$
15., (2), PROPOSITION A.26, (LTL-136)

provided that $\mu \oplus \varepsilon \leq \varepsilon_f$

q.e.d.

LEMMA 8.3 (FAILURE SAFETY) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check

time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$
\begin{array}{lll}
\Sigma_{GMP} & \vdash_{\text{LTL}} & FH(GMP) \\
 & & \rightarrow \\
 & & [\ correct \wedge joined(g) \\
 & & \ \wedge \\
 & & \ \ \diamondsuit^{\text{L}}[\ \neg p\text{-}correct \wedge myid(p) \\
 & & \ \ \ \wedge \\
 & & \ \ \ \ [\neg p\text{-}correct\ \text{S}\ (failed(g') \wedge \boxminus^{\text{T}}(p \notin g \wedge myid(p)))] \\
 & & \ \ \ ] \\
 & & \rightarrow \\
 & & \ \ p \notin g \\
 & &]
\end{array}
$$

$\diamondsuit$

PROOF 8.3 (LEMMA 8.3) We prove the lemma by contradiction, that is:

$$
\begin{array}{lll}
\Sigma_{GMP} & \vdash_{\text{LTL}} & \neg[\ FH(GMP) \\
 & & \rightarrow \\
 & & \ \ [\ correct \wedge joined(g) \\
 & & \ \ \wedge \\
 & & \ \ \ \diamondsuit^{\text{L}}[\ \neg p\text{-}correct \wedge myid(p) \\
 & & \ \ \ \ \wedge \\
 & & \ \ \ \ \ [\neg p\text{-}correct\ \text{S}\ (failed(g') \wedge \boxminus^{\text{T}}(p \notin g \wedge myid(p)))] \\
 & & \ \ \ \ ] \\
 & & \ \ \rightarrow \\
 & & \ \ \ p \notin g \\
 & & \ \ ] \\
 & &] \rightarrow \bot
\end{array}
$$

1. $FH(GMP) \wedge correct \wedge joined(g)$

   $\wedge$

   $\diamondsuit^{\text{L}}\ [\ \neg p\text{-}correct \wedge myid(p)$

   $\wedge$

   $[\neg p\text{-}correct\ \text{S}\ (failed(g') \wedge \boxminus^{\text{T}}(p \notin g \wedge myid(p)))]$

   $]$

   $\wedge$

   $p \in g$ Assumption

2. $correct \wedge p \in g$ 1., (LTL-80), (LTL-136)

3. $\Diamond^{\mathrm{T}}(correct \wedge delivered(\mathrm{Pr}, g, p))$ 2., (32), (LTL-136)

4. $\Diamond^{\mathrm{T}}\Diamond^{\mathrm{LT}}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$ 3., (39), (LTL-136)

5. $\Diamond^{\mathrm{T}}\circledcirc^{\mathrm{L}}\Diamond^{\mathrm{T}}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
4., (LTL-73), (LTL-136)

6. $\circledcirc^{\mathrm{L}}\Diamond^{\mathrm{T}}\Diamond^{\mathrm{T}}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
5., PROPOSITION A.42, (LTL-136)

7. $\circledcirc^{\mathrm{L}}\Diamond^{\mathrm{T}}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
6., PROPOSITION A.32, (LTL-136)

8. $\circledcirc^{\mathrm{L}}[\ \neg p\text{-}correct \wedge myid(p)$
$\wedge$
$[\neg p\text{-}correct\ \mathrm{S}\ (failed(g') \wedge \boxdot^{\mathrm{T}}(p \notin g \wedge myid(p)))]$
$]$
1., (LTL-80), (LTL-136)

9. $\circledcirc^{\mathrm{L}}[\ \neg p\text{-}correct \wedge myid(p)$
$\wedge$
$\exists \delta : [\ \delta > 0$
$\wedge$
$\Diamond^{\mathrm{T}}_{=\delta}\ (failed(g') \wedge \boxdot^{\mathrm{T}}(p \notin g \wedge myid(p)))$
$\wedge$
$\boxminus^{\mathrm{T}}_{<\delta}\ \neg p\text{-}correct$
$]$
$]$ 8., (LTL-40), (LTL-41), (LTL-136)

10. $\circledcirc^{\mathrm{L}}[\ \neg initiated(\mathrm{Pr}, g, p) \wedge myid(p)$
$\wedge$
$\exists \delta : [\ \delta > 0$
$\wedge$
$\Diamond^{\mathrm{T}}_{=\delta}\ (failed(g') \wedge \boxdot^{\mathrm{T}}(p \notin g \wedge myid(p)))$

$\wedge$

$\boxminus^{\mathrm{T}}_{<\delta}\ \neg initiated(\mathrm{Pr}, g, p)$

]

] 9., (23), (LTL-80), (LTL-136)

11. $\diamond^{\mathrm{L}}[\ \neg initiated(\mathrm{Pr}, g, p) \wedge myid(p)$

$\wedge$

$\exists\delta : [\ \delta > 0$

$\wedge$

$\diamondminus^{\mathrm{T}}_{=\delta}\ (failed(g') \wedge \boxdot^{\mathrm{T}}\boxdot^{\mathrm{T}}\ (\neg initiated(\mathrm{Pr}, g, p)))$

$\wedge$

$\boxminus^{\mathrm{T}}_{<\delta}\ \neg initiated(\mathrm{Pr}, g, p)$

]

] 10., (34), (LTL-136)

12. $\diamond^{\mathrm{L}}[\ \neg initiated(\mathrm{Pr}, g, p)$

$\wedge$

$\exists\delta : [\ \delta > 0$

$\wedge$

$\diamondminus^{\mathrm{T}}_{=\delta}\ \boxdot^{\mathrm{T}}(\neg initiated(\mathrm{Pr}, g, p))$

$\wedge$

$\boxminus^{\mathrm{T}}_{<\delta}\ \neg initiated(\mathrm{Pr}, g, p)$

]

] 11., (LTL-44), (LTL-45) (LTL-80), (LTL-136)

13. $\diamond^{\mathrm{L}}[\ \neg initiated(\mathrm{Pr}, g, p)$

$\wedge$

$\exists\delta : [\ \delta > 0$

$\wedge$

$\boxdot^{\mathrm{T}}_{\geq\delta}\ (\neg initiated(\mathrm{Pr}, g, p))$

$\wedge$

$\boxminus^{\mathrm{T}}_{<\delta}\ \neg initiated(\mathrm{Pr}, g, p)$

]

] 12., PROPOSITION A.4, (LTL-136)

14. $\diamondsuit^{\mathrm{L}}[\neg initiated(\mathrm{Pr}, g, p) \wedge \boxminus^{\mathrm{T}}(\neg initiated(\mathrm{Pr}, g, p))]$
13., PROPOSITION A.6, (LTL-136)

15. $\diamondsuit^{\mathrm{L}} \boxminus^{\mathrm{T}}(\neg initiated(\mathrm{Pr}, g, p))$
14., PROPOSITION A.5, (LTL-136)

16. $\bot$ 7., 15., (LTL-136)

Because the negation of the original formula leads to false the orginal formula itself must be true. *q.e.d.*

LEMMA 8.4 (LOCAL GROUP MEMBERSHIP) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\mathrm{LTL}} FH(GMP) \rightarrow [\, correct \wedge joined(g) \wedge myid(p) \rightarrow p \in g \,]$$

$\diamondsuit$

PROOF 8.4 (LEMMA 8.4) We prove the lemma by contradiction, that is:

$$\Sigma_{GMP} \vdash_{\mathrm{LTL}} \neg [\, FH(GMP) \rightarrow [\, correct \wedge joined(g) \wedge myid(p) \wedge p \notin g \,] \rightarrow \bot$$

1. $correct \wedge joined(g) \wedge myid(p) \wedge p \notin g$ Assumption
2. $correct \wedge joined(g) \wedge myid(p)$ 1., (LTL-80), (LTL-136)
3. $\diamondminus^{\mathrm{T}}(correct \wedge delivered(\mathrm{Pr}, g, p))$ 2., (49), (LTL-136)
4. $correct \wedge p \notin g$ 1., (LTL-80), (LTL-136)
5. $\neg \diamondminus^{\mathrm{T}}(correct \wedge delivered(\mathrm{Pr}, g, p)$ 4., (33), (LTL-136)
6. $\bot$ 3., 5., (LTL-136)

Because the negation of the original formula leads to false the orginal formula itself must be true. *q.e.d.*

LEMMA 8.5 (LOCAL STABILITY) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ correct \wedge joined(g) \rightarrow (correct \wedge joined(g))\ \mathrm{U}\ [\ failed(g) \vee \exists g' : (correct \wedge g \neq g' \wedge joined(g'))\ ]\ ]$$

◇

PROOF 8.5 (LEMMA 8.5)

1. $correct \wedge joined(g)$ — Assumption
2. $\exists p : (correct \wedge joined(g) \wedge myid(p))$ — 1., (41), (LTL-136)
3. $\exists p : (correct \wedge p \in g \wedge myid(p))$ — 2., (51), (LTL-136)
4. $\exists p : [\ (correct \wedge p \in g)\ \mathrm{U}\ (failed(g) \vee \exists m, g', p' : (\ correct \wedge g \neq g' \wedge delivered(m, g', p') \wedge myid(p')\ )\ )\ ]$ — 3., (36), (LTL-136)

5. $\exists p : [\ (correct \wedge joined(g))$
   U
   $(failed(g) \vee \exists m, g', p' : (\ correct$
   $\wedge\ g \neq g'$
   $\wedge\ delivered(m, g', p')$
   $\wedge\ myid(p')$
   $)$
   $)$
   $]$ 4., (50), (LTL-136)

6. $\exists p : [\ (correct \wedge joined(g))$
   U
   $(failed(g) \vee \exists g' : (correct \wedge g \neq g' \wedge joined(g')))$
   $]$ 5., (48), (LTL-83), (LTL-136)

7. $(correct \wedge joined(g))\ \mathrm{U}\ (failed(g) \vee \exists g' : (correct \wedge g \neq g' \wedge joined(g')))$
   6., (LTL-83), (LTL-136)

q.e.d.

LEMMA 8.6 (AGREEMENT ON GROUP MEMBERSHIP) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\mathrm{LTL}} FH(GMP) \rightarrow [\ correct \wedge joined(g) \wedge p \in g \rightarrow \neg\nabla^{L}(correct \wedge joined(g) \wedge p \notin g))\ ]$$

◇

PROOF 8.6 (LEMMA 8.6) We prove the lemma by contradiction, that is:

$$\Sigma_{GMP} \vdash_{\mathrm{LTL}} \neg[\ FH(GMP) \rightarrow [\ correct \wedge joined(g) \wedge p \in g \rightarrow \nabla^{L}(correct \wedge joined(g) \wedge p \notin g))\ ]\ ] \rightarrow \bot$$

1. $FH(GMP) \wedge correct \wedge joined(g) \wedge p \in g \wedge \nabla^{L}(correct \wedge joined(g) \wedge p \notin g)$
   Assumption
2. $correct \wedge p \in g$ 1., (LTL-80), (LTL-136)
3. $\Diamond^{T}(correct \wedge delivered(\mathrm{Pr}, g, p))$ 2., (32), (LTL-136)
4. $\Diamond^{T}\Diamond^{LT}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$ 3., (39), (LTL-136)
5. $\Diamond^{L}\Diamond^{T}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
   4., PROPOSITION A.42, PROPOSITION A.32, (LTL-136)
6. $\nabla^{L}(correct \wedge p \notin g)$ 1., (LTL-80), (LTL-136)
7. $\nabla^{L}\neg\Diamond^{T}(correct \wedge delivered(\mathrm{Pr}, g, p))$ 6., (33), (LTL-136)
8. $\nabla^{L}\neg\Diamond^{T}\Diamond^{LT}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$ 7., (39), (LTL-136)
9. $\nabla^{L}\neg\Diamond^{LT}(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
   8., PROPOSITION A.42, PROPOSITION A.32, (LTL-136)
10. $\nabla^{L}\boxdot^{L}\boxminus^{T}\neg(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
    9., PROPOSITION A.20, PROPOSITION A.39, (LTL-136)
11. $\boxdot^{L}\boxminus^{T}\neg(correct \wedge initiated(\mathrm{Pr}, g, p) \wedge myid(p))$
    10., PROPOSITION A.38, (LTL-136)
12. $\bot$ 5., 11., (LTL-136)

Because the negation of the original formula leads to false the orginal formula itself must be true. *q.e.d.*

LEMMA 8.7 (AGREEMENT ON GROUP ORDERING) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP. Let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check time period of the GMP and let $0 < \varepsilon$ be the diffusion time of

the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ [\ (correct \wedge joined(g)) \wedge \Diamond^{\text{T}} (correct \wedge joined(g'))\ ] \rightarrow \boxdot^{\text{L}} [\ \neg \boxbslash^{\text{T}} [\ (correct \wedge joined(g')) \wedge \Diamond^{\text{T}} (correct \wedge joined(g))\ ]\ ]\ ]$$

$\Diamond$

PROOF 8.7 (LEMMA 8.7)

1. $(correct \wedge joined(g)) \wedge \Diamond^{\text{T}} (correct \wedge joined(g'))$ — Assumption
2. $(correct \wedge joined(g) \wedge \exists p : myid(p)) \wedge \Diamond^{\text{T}}(correct \wedge joined(g') \wedge \exists p' : myid(p'))$ — 1., (41), (LTL-136)
3. $\exists p : [\ (correct \wedge joined(g) \wedge myid(p)) \wedge \Diamond^{\text{T}} (correct \wedge joined(g') \wedge myid(p))\ ]$ — 2., (42), (LTL-136)
4. $\exists p : \Diamond^{\text{T}} [\ (correct \wedge delivered(\text{Pr}, g, p) \wedge myid(p)) \wedge \Diamond^{\text{T}} (correct \wedge delivered(\text{Pr}, g', p) \wedge myid(p))\ ]$ — 3., (49), (LTL-80), (LTL-136)
5. $\exists p : \Diamond^{\text{T}} \boxdot^{\text{L}} [\ \neg \boxbslash^{\text{T}} [\ (correct \wedge delivered(\text{Pr}, g', p') \wedge myid(p')) \wedge \Diamond^{\text{T}} (correct \wedge delivered(\text{Pr}, g, p') \wedge myid(p'))\ ]\ ]$ — 4., (40), (LTL-136)

6. $\exists p : \boxdot^{\mathrm{L}} \diamondsuit^{\mathrm{T}} [\, \neg \boxbslash^{\mathrm{T}} [\, (correct \wedge delivered(\mathrm{Pr}, g', p') \wedge myid(p'))$
$\wedge$
$\diamondsuit^{\mathrm{T}} (correct \wedge delivered(\mathrm{Pr}, g, p') \wedge myid(p'))$
$]$
$]$ 5., PROPOSITION A.40, (LTL-136)

7. $\exists p : \boxdot^{\mathrm{L}} \diamondsuit^{\mathrm{T}} [\, \boxslash^{\mathrm{T}} \neg [\, (correct \wedge delivered(\mathrm{Pr}, g', p') \wedge myid(p'))$
$\wedge$
$\diamondsuit^{\mathrm{T}} (correct \wedge delivered(\mathrm{Pr}, g, p') \wedge myid(p'))$
$]$
$]$ 6., PROPOSITION A.21, (LTL-136)

8. $\exists p : \boxdot^{\mathrm{L}} [\, \boxslash^{\mathrm{T}} \neg [\, (correct \wedge delivered(\mathrm{Pr}, g', p') \wedge myid(p'))$
$\wedge$
$\diamondsuit^{\mathrm{T}} (correct \wedge delivered(\mathrm{Pr}, g, p') \wedge myid(p'))$
$]$
$]$ 7., PROPOSITION A.33, (LTL-136)

9. $\exists p : \boxdot^{\mathrm{L}} [\, \neg \boxbslash^{\mathrm{T}} [\, (correct \wedge delivered(\mathrm{Pr}, g', p') \wedge myid(p'))$
$\wedge$
$\diamondsuit^{\mathrm{T}} (correct \wedge delivered(\mathrm{Pr}, g, p') \wedge myid(p'))$
$]$
$]$ 8., PROPOSITION A.21, (LTL-136)

10. $\boxdot^{\mathrm{L}} [\, \neg \boxbslash^{\mathrm{T}} [\, (correct \wedge joined(g'))$
$\wedge$
$\diamondsuit^{\mathrm{T}} (correct \wedge joined(g))$
$]$
$]$ 8., (48), (LTL-83), (LTL-136)

q.e.d.

LEMMA 8.8 (JOIN UNIQUENESS) Let Σ_{GMP} be the specification of the periodic broadcast membership protocol GMP and let $FH(GMP)$ be the conjunction of all fault-hypotheses for GMP. Furthermore, let $0 < \mu$ be the check

time period of the GMP and let $0 < \varepsilon$ be the diffusion time of the underlying atomic broadcast service.

$$\Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ correct \wedge joined(g) \wedge joined(g') \rightarrow g = g'\ ]$$

$\Diamond$

PROOF 8.8 (LEMMA 8.8)

1. $correct \wedge joined(g) \wedge joined(g')$ — Assumption
2. $correct \wedge joined(g) \wedge joined(g') \wedge \exists p : myid(p)$ — 1., (41), (LTL-136)
3. $\exists p : (correct \wedge myid(p) \wedge (p \in g) \wedge (p \in g')$ — 2., LEMMA 8.4, (LTL-136)
4. $\exists p : g = g'$ — 3., (35), (LTL-136)
5. $g = g'$ — 4., (LTL-83), (LTL-136)

q.e.d.

PROOF 8.9 (THEOREM 8.1) Recall that Σ_{GMP} is the specification of the periodic broadcast membership protocol GMP, that $0 < \mu$ is the check time period of the GMP, that $0 < \varepsilon$ is the diffusion time of the underlying atomic broadcast service, and that $FH(GMP)$ is the conjunction of all fault-hypotheses for GMP.

Because we have proved

- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ STARTUP LIVENESS — Lemma 8.1
- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ FAILURE LIVENESS — Lemma 8.2
- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ FAILURE SAFETY — Lemma 8.3
- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ LOCAL GROUP MEMBERSHIP — Lemma 8.4
- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ LOCAL STABILITY — Lemma 8.5
- $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ AGREEMENT ON GROUP MEMBERSHIP — Lemma 8.6

▷ $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ AGREEMENT ON GROUP ORDERING Lemma 8.7

▷ $\Sigma_{GMP} \vdash FH(GMP) \rightarrow$ JOIN UNIQUENESS Lemma 8.8

and with the scheme

$$[(\varphi \rightarrow \varphi_1) \wedge (\varphi \rightarrow \varphi_2)] \leftrightarrow [\varphi \rightarrow (\varphi_1 \wedge \varphi_2)]$$

of propositional logic we can directly derive:

$$\begin{aligned} \Sigma_{GMP} \vdash_{\text{LTL}} FH(GMP) \rightarrow [\ & \text{STARTUP LIVENESS} \\ & \wedge \text{FAILURE LIVENESS} \\ & \wedge \text{FAILURE SAFETY} \\ & \wedge \text{LOCAL GROUP MEMBERSHIP} \\ & \wedge \text{LOCAL STABILITY} \\ & \wedge \text{AGREEMENT ON GROUP MEMBERSHIP} \\ & \wedge \text{AGREEMENT ON GROUP ORDERING} \\ & \wedge \text{JOIN UNIQUENESS} \\ &] \end{aligned}$$

q.e.d.

8.4 SOME CONCLUSIONS

Compared to the original work by Cristian [19] and Cristian et al. [22] we have presented more rigorous specifications for both the processor group membership service and a class of protocols. All static properties, whether local or distributed, have been left informal in [19] as well as in [22]. For example, properties about the network such as the absence of network partitions have been provided there only informally. Moreover, formal proofs of the correctness of the protocols are completely missing there. Hence, we may conclude that our specifications and proofs will be more rigorous than those given by Cristian et al.

Another distinction is again (cf. chapter 7) given by the formal method applied: in [19] and in [22], it is first-order predicate logic for the specification of the service and a particular Pascal-like programming language used by Cristian et al. whereas we made use of our locative temporal logic for both the service specification as well as the protocol specification. Although the use of a particular programming language would be better suited for verification

purposes a strict formal verification approach, e.g., a formal semantics of the programming language will also be missing in [19] and [22].

So, we mean that this chapter has shown again (cf. chapter 7) the usefulness of our two-sorted approach for the specification and verification of systems that are more complex than those discussed in the beginning of this part II. In particular, we have seen that most of the work in proving static and dynamic properties can be done locally. The properties of the atomic broadcast service can then be used to prove distributed properties. Moreover, it is, also in this chapter, important to take care that all GMS and GMP properties were of relative nature, that is, all these properties express nothing about the existence of locations *and* time points where a particular view on processor group membership must have really become valid. It depends on the correctness of the underlying dynamically evolving network.

CHAPTER 9

SUMMARY

OUTLINE

This chapter summarizes our investigations presented in part I and part II of this thesis. In particular, the general properties (cf. chapter 1) that one should require of an adequate specification method will be examined in the context of our specification method LTL. Still open points within our logical system LTL will be reviewed and possible directions for future work will be discussed.

This chapter contains the following sections: Section 9.1, provides a summary of our investigations undertaken in this thesis and draws some conclusions. Section 9.2 briefly reviews still open points within LTL and introduces some possible items for future work.

9.1 SUMMARY AND CONCLUSIONS

In this thesis, we have been concerned with a new formal method for the specification of distributed real-time systems. This method, called locative temporal logic (LTL), has been built from two different modal logics: a temporal logic to express temporal properties and a locative logic to express locative properties. The underlying model has been defined as a two-sorted Kripke-model in the sense that temporal and locative universes have been combined to form a direct product. Binary relations of locative and temporal accessibility on the product space have been provided on the semantics level. Moreover, we have added suitable distance functions and metric domains for the temporal and locative sorts.

Suitable modal operators, i.e., locative temporal connectives have been incorporated in the logical language. The language itself is a first-order language where quantification over the locative temporal universe is disallowed but quantification over the corresponding metric domain is allowed. The locative temporal operators have been made metric because properties in distributed real-time systems ask for distances in space and time (cf. chapter 1 and chapter 6).

We have applied our specification method to a number of paradigms from the field of distributed real-time systems: *dining philosophers*, *distributed watchdog*, *point-to-point-based* and *diffusion-based communication*, *synchronous* and *asynchronous communication*, *atomic broadcast*, and *processor group membership*. The dining philosophers paradigm has been studied from a theoretical point of interest: to become able to compare several kinds of logics and to demonstrate the suitability of our approach we have provided specifications of a community of dining philosphers in first-order predicate logic, in temporal logic, in locative logic, and in locative temporal logic. All other paradigms have been studied from a practical point of interest: will our formal method be *adequate* for the specification of distributed real-time systems?

The logical system LTL is, at least to a certain extent, a solution of the structural defect as mentioned in section 1.1.3, that is, a solution of the problem in classical logic-based specification formalisms that no distinction can be made explicit between local and global properties of a distributed real-time system. Our paradigm of an *external observer in space and time* (cf. section 5.1) exactly allows to distinguish between local, distributed, static, and dynamic properties.

To give a more detailed answer to the above question recall from chapter 1 the general properties that should be required of an adequate specification method: *formality*, *abstractness*, *expressiveness*, *separation of concerns*, and

unity within diversity. Abstractness and expressiveness together determine *what* adequacy is and separation of concerns and unity within diversity together determine *how* adequacy can be reached.

9.1.1 FORMALITY

Formality is a commonly accepted property of specification methods. It ensures concise and unambiguous specifications and rigorous proofs. Without formality specifications, in general, become strongly dependent on the author's and reader's intuitions and insights into the system at issue. Moreover, the contribution of formality of a specification method during the development of a particular system is at least two-fold: firstly, formality allows for formal *verification* of system properties and, secondly, formality helps to get an adequate view on the system itself (*validation*).[1]

Our specification method LTL as introduced and applied in this thesis is a logical system comprising a formal language together with a formal semantics for the specification of properties of distributed real-time systems and a deduction system for reasoning about such properties (cf. chapter 4 and appendix A). Deduction in LTL is founded on a binary consequence relation expressing the fact that "the formula on the right hand side is derivable w.r.t. space and time from the formula (or set of formulae) on the left hand side of the relation symbol".

Although formal proofs of soundness and completeness of LTL were missing here we have no reasons to assume that these properties remain to be missing. This is, firstly, because of corresponding results for temporal logics with D-operator by Koymans [54] and de Rijke [76], secondly, because our locative logic is an S5-logic (cf. chapter 3) for which corresponding results do exist (e.g. Chellas [15]), and, thirdly, because of soundness and completeness results for many-sorted (many-dimensional) modal logics by Stuhlmann-Laeisz [83] and Venema [87]. Soundness and completeness properties for our logical system LTL are under investigation in a joint work with Wiebe van der Hoek (cf. [91]).

[1] Although validation cannot completely be formalized—because it deals with the relation between reality and formal specification—formality of our specification method has helped us to find discrepancies between the informal original statement and our intuition in the early stages of formalization and the formal specifications. This was especially useful in chapter 7 on atomic broadcast and in chapter 8 on processor group membership. Of course, this benefit could not become apparent because we were not concerned, in this thesis, with the development of the several specifications and proofs but with their end products only.

9.1.2 ABSTRACTNESS

Abstractness is also a commonly accepted property of specification methods that ensures that *irrelevant details* must not be specified. Implementation bias must not be considered if it is not required at the chosen level of abstraction.

In section 1.2, we have identified locative and temporal properties to be common to all distributed real-time systems. Locative properties are manifested themselves in the distribution of processes "among spatially separated autonomous processors". Temporal properties become relevant in the sense that processes "have to adhere to timing constraints". These two principal classes of properties will be independent of any particular "overall goal" that has to be achieved by a particular system.

Our specification method LTL as introduced and applied in this thesis provides for modal logical operators, i.e., locative temporal connectives that can be used to specify locative (structural) and temporal (behavioural) properties of distributed real-time systems independent of concrete implementations of the structure and behaviour. This has become apparent during our discussions of the dining philosophers paradigm: where other logical specifications were, to some extent, depending on the number of philosophers—at least by some quantification on the philosophers domain—our specification has been freed of such bias although only propositional variables have been used. For example, the approach taken by Barringer et al. in [6] would ask for additional specification formulae when adding philosophers to the community or removing philosophers from it. In our specification, such modifications would not have any impact (cf. section 4.4).

9.1.3 EXPRESSIVENESS

Expressiveness is a further commonly accepted property of specification methods that ensures that all *relevant details* of the application area can be specified.

As we have seen in chapter 1 locative and temporal properties must at least be expressible by a formal method that is to be regarded suitable for the specification of distributed real-time systems.

Our specification method LTL as introduced and applied in this thesis provides for modal logical operators, i.e., locative temporal connectives that can be used to specify locative (structural) and temporal (behavioural) properties of distributed real-time systems. In part II, we have demonstrated that our logical system can be applied not only to theoretically convenient examples (cf. running example in part I and chapters 5 and 6) but also to more complex

paradigms.

In chapter 7, we have discussed the problem of broadcasting messages in a communication network that is sensitive to particular faults. The paradigm of an atomic broadcast service has been taken from [20]. We have provided service as well as protocol specifications and we have proved that the protocol specification is a specification refinement of the service specification, that is, we have proved that the properties of the specified class of protocols allow to derive the properties of the atomic broadcast service within LTL.

In chapter 8, we have discussed the problem of reaching agreement on the view of processor group membership in a distributed real-time system where failing and (re-) starting processors have been taken into account. The paradigm of processor group membership service has been taken from [22]. We have again provided service as well as protocol specifications and we have also proved that the protocol specification is a specification refinement of the service specification. The underlying idea of the class of protocols has been given by the fact that the correctly functioning processors periodically broadcast messages to all others in the network. Thereby, we could make use of a slightly modified version of the service specification of the atomic broadcast paradigm.

9.1.4 SEPARATION OF CONCERNS

Separation of concerns is, in fact, a structural property and really means "a priori fixed separation of concerns explicitly visible in the specification method". It highly depends on the application area and can be achieved in many different ways. Separation of concerns supports structured specifications enforcing a clear understanding of the necessary details.

Our specification method LTL as introduced and applied in this thesis provides for modal logical operators, i.e., locative temporal connectives that allow for syntactically characteristic formulae of four classes of properties (cf. section 4.1.1): *local static* properties which are neither evolving in space nor in time, *local dynamic* properties which are not evolving in space but in time, *distributed static* properties which are not evolving in time but in space, and finally *distributed dynamic* properties which are evolving in space and time. All such properties have been considered not only from a qualitative but also from a quantitative point of view.

9.1.5 UNITY WITHIN DIVERSITY

Unity within diversity is, in fact, again a structural property and really means "a priori fixed unity within diversity explicitly visible in the specification method". It is complementary to separation of concerns in the sense that a relationship must be established between the separated, seemingly diverse concerns. As separation of concerns it also highly depends on the application area and can be achieved in many different ways.

Unification in logic-based specification formalisms is usually achieved by a suitable consequence relation in the corresponding deduction system. For example, in temporal logics logical deduction is given by temporal deduction.

Our specification method LTL as introduced and applied in this thesis provides a logical consequence relation that is dependent on space and time, that is, our binary consequence relation "$\Sigma \vdash_{LTL} \phi$" expresses the fact that "formula φ is derivable w.r.t. space and time from the set of formulae Σ" (cf. chapter 4 and appendix A). In this respect, we can say with Koymans [54] that our "specification method is based on a single formalism covering all aspects of a specification" of distributed real-time systems and that "the specifications remain purely locative temporal".

9.2 FUTURE WORK

Of course, there is still much work left for future investigations to get more insight into locative temporal logic, its metricated version, and its applicability to the specification and verification of distributed real-time systems. But we mean that we have demonstrated that LTL is a powerful and suitable tool for such tasks and that it resolves the structural defect as mentioned in chapter 1. Let us discuss some possible topics for future research in more detail.

First of all, we shall need formal proofs of soundness and completeness of our logical system LTL that are missing in this thesis (cf. chapter 4). Such proofs are under investigation in a joint work with Wiebe van der Hoek [91].

LOCATIVE LOGIC

In this thesis, locative logic (cf. chapter 3) has been introduced only for special purposes, i.e., purposes where an equivalence relation will be useful as locative reachability relation. It might be interesting to relax the list of properties for the locative reachability relation.

For example, giving up the axiom of symmetry and using irreflexive locative operators would lead to formulae that are characteristic for certain types of

nodes in a graph (cf. Harary et al. [36] for more information on graphs). An illustration is given in table 9.1.

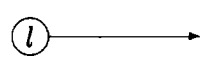
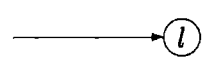
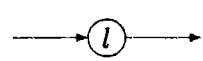
	(Sub-)Graph	LTL-formula
Source		$\neg\Diamond^{L}\top$
Sink		$\neg\oplus^{L}\top$
Transmitter		$\Diamond^{L}\top \wedge \oplus^{L}\top$
Isolate		$\neg(\Diamond^{L}\top \vee \oplus^{L}\top)$

Table 9.1: Sources, Sinks, Transmitters, and Isolates

Location l is called a *source* if there exists no location, except l itself, from which it is reachable (see first row). This can be specified in LTL by making use of the locative operator $\Diamond^{L}$ (somewhere in the back).

Location l is called a *sink* if there exists no location, except l itself, that is reachable from it (see second row). This can be specified in LTL by making use of the locative operator $\oplus^{L}$ (somewhere in the front).

Combining the characteristic properties of sources and sinks we get a characteristic property of a transmitter. Location l is called a *transmitter* if it is reachable from at least one location as well as it can reach itself at least one location (see third row).

A location l is called an *isolate* if there exists no location, except l itself, that is either reachable from it or from which l can be reached. This can be specified in LTL by combining the formulae for sources and sinks but now with disjunction and negation (see fourth row).

SPECIFICATION AND VERIFICATION TOOL

Recall that we have confined ourselves, in this thesis, to specification issues in the context of distributed real-time system (cf. preface). Following ideas which we have presented in [90] it will be interesting to further investigate the applicability of LTL for the specification and verification of distributed real-time systems. For example, what kind of impacts will a specification language based on LTL have on a formal tool consisting of a programming language, a specification language, and a compositional proof system such as introduced by Hooman [43]? Do we get simpler proof rules for parallel composition because true parallelism is in our approach the "normal case" and sequential composition is the "special case"? Or would it be inconvenient to use LTL because decisions about dispersion of processes among processors have to be taken too early in the development process? Recall from section 1.2 the rule of "general parallel composition" (cf. also Hooman [43]):

$$\frac{\pi_1 \text{ SAT } \varphi_1 \ , \ \pi_2 \text{ SAT } \varphi_2}{\pi_1 \parallel \pi_2 \text{ SAT } (\varphi_1 \wedge (\varphi_2 \mathrm{C} \, \Box \, \mathit{noact}(\mathit{dch}(\pi_2)))) \vee (\varphi_2 \wedge (\varphi_1 \mathrm{C} \, \Box \, \mathit{noact}(\mathit{dch}(\pi_1))))}$$

This quite complex rule was necessary in the general case for reasons of completeness of the proof system because φ_1 and φ_2 were allowed to contain the termination predicate *done*. As we have seen in the previous chapters our locative temporal logic allows to distinguish between truth of the predicate *done* at a certain point in time at one location and its truth at a certain point in time at other locations without using more than one such predicate. So we have good reasons to assume that our two-sorted approach will lead to a proof rule for general parallel composition that is as simple and intuitive as the rule of "simple parallel composition" (cf. section 1.2). Presupposing that π_1 and π_2 are dispersed among different processors such a rule could look like the following:

$$\frac{\pi_1 \text{ SAT } \varphi_1 \ , \ \pi_2 \text{ SAT } \varphi_2}{\pi_1 \parallel \pi_2 \text{ SAT } \varphi_1 \wedge \varphi_2}$$

As a drawback of such simplication in the general case another problem could arise for a classically simpler case: what should we do with processes that are running on the same processor, that is, where the parallel composition of such processes follows an interleaving semantics such as given by the |||-operator in CSP [42]?

CLOCK SYNCHRONIZATION

Another interesting problem from the field of distributed real-time systems is that of clock synchronization in a distributed clock system. This is basically

the problem of computing adjustments to hardware clocks and to maintain the corresponding logical clocks. To deal with unsynchronized (hardware) clocks a particular service is needed (Cristian et al. [18]): A synchronized clocks service is a distributed service that enables the correct processors in a distributed real-time system to provide the same view on time. Providing *exactly* the same view on time at different processors in case of a distributed clock system would be unrealistic (cf. chapter 5). The way out of this dilemma is to have the clocks *approximately* synchronized, i.e., within some a priori known bounds. It will be interesting to take the problem of clock synchronization as a further non-trivial example and to provide service and protocol specifications in LTL. Doing so would lead, in restrospect on chapter 3, to a justification of our decision to distinguish between meta-level time notion and object-level time notion (cf. section 8.2 where the analogue has been stated for our space notions).

AUTOMATED TOOL SUPPORT

Another important feature for the acceptability of formal specification methods has become more prominent in the last few years: automated tool support for the development of distributed real-time systems. In particular, it will be interesting to consider problems of state space explosion during the verification of such complex systems: can we get rid of these problems or can we at least decrease their consequences when using our logical system LTL?

Appendix A

The Logical System LTL

Outline

In general, a logical system consists of the *set of well-formed formulae* given through the language of the logic and the notion of *consequence*, that is, "a particular formula is derivable from a set of formulae in the corresponding logic". The logical system for LTL to be presented here has been used throughout the whole part II of this thesis.

This appendix contains the following sections: In section A.1, we will summarize the language of LTL resulting from the investigations in part I and used in part II. Section A.2 provides the formal semantics of basic logical operators of LTL. Section A.3 contains the definition of LTL consequence, that is, definitions of dual and other non-primitive operators, axioms, and rules. At the end we will state some properties derivable within LTL.

A.1 LANGUAGE

The language of metric locative temporal logic as used in part II is, in fact, a metricated polymodal language $PML(R^{\mathrm{U}}_{\mathrm{L}}, R^{\mathrm{U}}_{\mathrm{T}}, R^{\approx}_{\mathrm{L}}, R^{<}_{\mathrm{T}}, =_{\mathrm{L}}, =_{\mathrm{T}}, \neq_{\mathrm{L}}, \neq_{\mathrm{T}})$ with the universal locative relation $R^{\mathrm{U}}_{\mathrm{L}}$ on $L \times T$, the universal temporal relation $R^{\mathrm{U}}_{\mathrm{T}}$ on $L \times T$, the locative acessibility relation $R^{\approx}_{\mathrm{L}}$ on $L \times T$, the temporal acessibility relation $R^{<}_{\mathrm{T}}$ on $L \times T$, the relation $=_{\mathrm{L}}$ of loative equality on $L \times T$, the relation $=_{\mathrm{T}}$ of temporal equality on $L \times T$, the relation $\neq_{\mathrm{L}}$ of locative inequality on $L \times T$, and the relation $\neq_{\mathrm{T}}$ of temporal inequality on $L \times T$.

Metric locative temporal logic as used in part II is a propositional version w.r.t. the temporal and locative sorts, that is, quantification over the set of time points and the set of locations is forbidden. But we allow for quantification over the metric domains $\mathbb{D}_{\mathrm{L}}$ and $\mathbb{D}_{\mathrm{T}}$. For other sorts such as, e.g., a message domain (cf. chapter 7) or a group domain (cf. chapter 8) we have been making use of first-order predicate logic. The logical system of first-order predicate logic is standard (cf. part I and, e.g., Chang and Keisler [14]).

SYMBOLS The following symbols will be used in metric locative temporal logic thereby neglecting any difference between syntax and semantics of elements from the metric locative and temporal domains:

1. *constant temporal distance symbols*: an enumerable set $\mathbb{D}_{\mathrm{T}}$ of constant symbols including 0 (for simplicity, we assume $\mathbb{D}_{\mathrm{T}} = \mathbb{Q}^{+}_{0}$, that is, the set of non-negative rational numbers including 0)
2. *constant locative distance symbols*: an enumerable set $\mathbb{D}_{\mathrm{L}}$ of constant symbols including 0 (for simplicity, we assume $\mathbb{D}_{\mathrm{L}} = \mathbb{N}_0$, that is, the set of natural numbers including 0)
3. *variable temporal distance symbols*: an enumerable set $\mathcal{A}_{V_T}$ of variable symbols
4. *variable locative distance symbols*: an enumerable set $\mathcal{A}_{V_L}$ of variables symbols
5. *operation symbols*:[1] infix function symbols $\oplus$, $\otimes$, $=$, and $<$ all of arity 2
6. *propositional variables*: an enumerable set $\mathcal{A}_P$ of constant predicate symbols

[1]Note that the operator $\oplus$ is used for addition between locative distances as well as for addition between temporal distances. Similarly, the operators $=$ and $<$ are both used between locative terms and temporal terms. Thus, all three symbols will be used here in an overloaded manner.

7. *propositional connectives*: $\top$, $\bot$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
8. *quantifier symbols*: $\forall$, $\exists$,
9. *locative temporal connectives*:
   $\boxdot^{L}$, $\diamondsuit^{L}$, $\triangle^{L}$, $\triangledown^{L}$, $\blacktriangle^{L}$, $\blacktriangledown^{L}$,
   $\boxminus^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\oplus^{T}$, $\circledcirc^{T}$, $\ominus^{T}$, $\circledast^{T}$, $\oplus^{T}$, $\triangle^{T}$, $\triangledown^{T}$, $\blacktriangle^{T}$, $\blacktriangledown^{T}$, S, U,
   $\boxminus^{LT}$, $\diamondsuit^{LT}$, $\boxplus^{LT}$, $\oplus^{LT}$, $\circledcirc^{LT}$, $\ominus^{LT}$, $\circledast^{LT}$, $\oplus^{LT}$, $\triangle^{LT}$, $\triangledown^{LT}$, $\blacktriangle^{LT}$, $\blacktriangledown^{LT}$

PRIORITIES The priorities of the propositional and locative temporal connectives are defined as usual in the following order (within one item the priorities are the same):

1. $\neg$,
   $\boxdot^{L}$, $\diamondsuit^{L}$, $\triangle^{L}$, $\triangledown^{L}$, $\blacktriangle^{L}$, $\blacktriangledown^{L}$
   $\boxminus^{T}$, $\diamondsuit^{T}$, $\boxplus^{T}$, $\oplus^{T}$, $\circledcirc^{T}$, $\ominus^{T}$, $\circledast^{T}$, $\oplus^{T}$, $\triangle^{T}$, $\triangledown^{T}$, $\blacktriangle^{T}$, $\blacktriangledown^{T}$,
   $\boxminus^{LT}$, $\diamondsuit^{LT}$, $\boxplus^{LT}$, $\oplus^{LT}$, $\circledcirc^{LT}$, $\ominus^{LT}$, $\circledast^{LT}$, $\oplus^{LT}$, $\triangle^{LT}$, $\triangledown^{LT}$, $\blacktriangle^{LT}$, $\blacktriangledown^{LT}$ (highest priority)
2. S, U
3. $\wedge$, $\vee$
4. $\rightarrow$, $\leftrightarrow$ (lowest priority)

WELL-FORMED LOCATIVE TERMS The set of well-formed locative terms in metric locative temporal logic is the minimal set of terms closed under the following formation rules:

1. A constant locative distance symbol is a well-formed locative term.
2. A variable locative distance symbol is a well-formed locative term.
3. If η_1 and η_2 are well-formed locative terms then $\eta_1 \oplus \eta_2$ is a well-formed locative term.

WELL-FORMED TEMPORAL TERMS The set of well-formed temporal terms in metric locative temporal logic is the minimal set of terms closed under the following formation rules:

1. A constant temporal distance symbol is a well-formed temporal term.
2. A variable temporal distance symbol is a well-formed temporal term.
3. If τ_1 and τ_2 are well-formed temporal terms then $\tau_1 \oplus \tau_2$ is a well-formed temporal term.

WELL-FORMED FORMULAE The set of well-formed formulae in metric locative temporal logic is the minimal set of formulae closed under the following formation rules:

1. A propositional variable is a well-formed formula.
2. The propositional connective $\bot$ (false) is a well-formed formula.
3. If φ_1 and φ_2 are well-formed formulae then $\varphi_1 \rightarrow \varphi_2$ is a well-formed formula.
4. If η_1 and η_2 are well-formed locative terms then $\eta_1 = \eta_2$ and $\eta_1 < \eta_2$ are well-formed formula.
5. If τ_1 and τ_2 are well-formed temporal terms then $\tau_1 = \tau_2$ and $\tau_1 < \tau_2$ are well-formed formula.
6. If η is a well-formed locative term and φ is a well-formed formula then $\boxdot^{\mathsf{L}}_{=\eta}\,\varphi$, $\Delta^{\mathsf{L}}_{=\eta}\,\varphi$, and $\diamondsuit^{\mathsf{L}}_{=\eta}\,\varphi$ are well-formed formulae.
7. If τ is a well-formed temporal term and φ is a well-formed formula then $\boxminus^{\mathsf{T}}_{=\tau}\,\varphi$, $\boxplus^{\mathsf{T}}_{=\tau}\,\varphi$, $\Delta^{\mathsf{T}}_{=\tau}\,\varphi$, and $\diamondsuit^{\mathsf{T}}_{=\tau}\,\varphi$ are well-formed formulae.
8. If ζ is a variable locative distance symbol and φ is a well-formed formula then $\forall\zeta : \varphi$ is a well-formed formula.
9. If δ is a variable temporal distance symbol and φ is a well-formed formula then $\forall\delta : \varphi$ is a well-formed formula.

Note that all non-primitive operators will be defined in section A.3 below.

A.2 FORMAL SEMANTICS

The formal semantics of basic logical operators in metric locative temporal logic is founded on the formal notion of locative temporal satisfiability. Let us presuppose a non-empty set L of locations, a non-empty set T of time points, a set Σ_L of well-formed locative terms, and a set Σ_T of well-formed temporal

terms. Furthermore, we assume a metric locative temporal model, i.e.,

$$\mathcal{M}^{|\cdot|}_{\mathrm{LT}} = \left\{ \begin{array}{ll} (L \times T & \text{locative temporal universe} \\ , \mathbb{D}_{\mathrm{L}} & \text{locative metric domain} \\ , \mathbb{D}_{\mathrm{T}} & \text{temporal metric domain} \\ , R^{\approx}_{\mathrm{L}} \subseteq (L \times T) \times (L \times T) & \text{locative reachability} \\ , R^{\mathrm{U}}_{\mathrm{L}} \subseteq (L \times T) \times (L \times T) & \text{locative universal relation} \\ , R^{<}_{\mathrm{T}} \subseteq (L \times T) \times (L \times T) & \text{temporal precedence} \\ , R^{\mathrm{U}}_{\mathrm{T}} \subseteq (L \times T) \times (L \times T) & \text{temporal universal relation} \\ , \rho_{\mathrm{L}} : L \times L \longrightarrow \mathbb{D}_{\mathrm{L}} & \text{locative distance function} \\ , \rho_{\mathrm{T}} : T \times T \longrightarrow \mathbb{D}_{\mathrm{T}} & \text{temporal distance function} \\ , \oplus_{\mathrm{L}} : (\mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}}) \longrightarrow \mathbb{D}_{\mathrm{L}} & \text{locative distance addition} \\ , \oplus_{\mathrm{T}} : (\mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}}) \longrightarrow \mathbb{D}_{\mathrm{T}} & \text{temporal distance addition} \\ , 0 \in \mathbb{D}_{\mathrm{L}} & \text{unity element w.r.t. } \oplus_{\mathrm{L}} \\ , 0 \in \mathbb{D}_{\mathrm{T}} & \text{unity element w.r.t. } \oplus_{\mathrm{T}} \\ , =\subseteq \mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}} & \text{equality on } \mathbb{D}_{\mathrm{L}} \\ , =\subseteq \mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}} & \text{equality on } \mathbb{D}_{\mathrm{T}} \\ , \neq\subseteq \mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}} & \text{inequality on } \mathbb{D}_{\mathrm{L}} \\ , \neq\subseteq \mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}} & \text{inequality on } \mathbb{D}_{\mathrm{T}} \\ , <\subseteq \mathbb{D}_{\mathrm{L}} \times \mathbb{D}_{\mathrm{L}} & \text{less-than on } \mathbb{D}_{\mathrm{L}} \\ , <\subseteq \mathbb{D}_{\mathrm{T}} \times \mathbb{D}_{\mathrm{T}} & \text{less-than on } \mathbb{D}_{\mathrm{T}} \\ , \alpha_{\mathrm{L}} : (L \times T) \times \Sigma_{\mathrm{L}} \longrightarrow \mathbb{D}_{\mathrm{L}} & \text{rigid locative binding} \\ , \alpha_{\mathrm{T}} : (L \times T) \times \Sigma_{\mathrm{T}} \longrightarrow \mathbb{D}_{\mathrm{T}} & \text{rigid temporal binding} \\ , \mathcal{I}_{\mathrm{LT}} : \mathcal{A}_P \longrightarrow \wp(L \times T) & \text{l-t-interpretation function} \\ ) & \end{array} \right.$$

Moreover, we assume a set $\mathcal{A}_P$ of propositional variables and a pair $\langle l, t \rangle \in (L \times T)$. Then, a well-formed formula φ *holds (is satisfied) in a metric locative temporal model* $\mathcal{M}^{|\cdot|}_{\mathrm{LT}}$ *at position* $\langle l, t \rangle$, notation $\mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \models_{\mathrm{LTL}} \varphi$, is defined inductively as follows:

FALSITY

$$\mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \not\models_{\mathrm{LTL}} \bot$$

ATOMIC PROPOSITION

$$\mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \models_{\mathrm{LTL}} p \quad \text{iff} \quad \langle l, t \rangle \in \mathcal{I}_{\mathrm{LT}}(p) \text{ for } p \in \mathcal{A}_P$$

IMPLICATION

$$\mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \models_{\mathrm{LTL}} \varphi_1 \rightarrow \varphi_2 \quad \text{iff} \quad \begin{array}{l} \text{if } \mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \models_{\mathrm{LTL}} \varphi_1 \\ \text{then } \mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t \rangle \models_{\mathrm{LTL}} \varphi_2 \end{array}$$

EQUALITY BETWEEN LOCATIVE TERMS

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \eta_1 = \eta_2 \quad \text{iff} \quad \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_1) = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_2)$$

LESS-THAN BETWEEN LOCATIVE TERMS

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \eta_1 < \eta_2 \quad \text{iff} \quad \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_1) < \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta_2)$$

EQUALITY BETWEEN TEMPORAL TERMS

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \tau_1 = \tau_2 \quad \text{iff} \quad \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_1) = \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_2)$$

LESS-THAN BETWEEN TEMPORAL TERMS

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \tau_1 < \tau_2 \quad \text{iff} \quad \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_1) < \alpha_{\mathrm{T}}(\langle l, t\rangle, \tau_2)$$

EVERYWHERE IN THE CLASS (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \varphi \quad \text{iff} \quad \begin{array}{l} \text{for all } \langle l', t\rangle \in (L \times T): \\ \text{if } R^{\approx}_{\mathrm{L}}\ \langle l, t\rangle\ \langle l', t\rangle \\ \quad \text{and } \rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}((l, t), \eta) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

EVERYWHERE ELSE (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \triangle^{\mathrm{L}}_{=\eta} \varphi \quad \text{iff} \quad \begin{array}{l} 0 \neq \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta) \\ \text{and} \\ \text{for all } \langle l', t\rangle \in (L \times T): \\ \text{if } \langle l, t\rangle \neq_{\mathrm{L}} \langle l', t\rangle \\ \quad \text{and } \rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

EVERYWHERE (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \triangleq^{\mathrm{L}}_{=\eta} \varphi \quad \text{iff} \quad \begin{array}{l} \text{for all } \langle l', t\rangle \in (L \times T): \\ \text{if } R^{\mathrm{U}}_{\mathrm{L}}\ \langle l, t\rangle\ \langle l', t\rangle \\ \quad \text{and } \rho_{\mathrm{L}}(l, l') = \alpha_{\mathrm{L}}(\langle l, t\rangle, \eta) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l', t\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

ALWAYS IN THE PAST (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t\rangle \models_{\mathrm{LTL}} \boxminus^{\mathrm{T}}_{=\tau}\, \varphi \quad \text{iff} \quad \begin{array}{l} 0 \neq \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{and} \\ \text{for all } \langle l,t'\rangle \in (L \times T): \\ \text{if } R^{\leq}_{\mathrm{T}}\ \langle l,t'\rangle\ \langle l,t\rangle \\ \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

ALWAYS IN THE FUTURE (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t\rangle \models_{\mathrm{LTL}} \boxplus^{\mathrm{T}}_{=\tau}\, \varphi \quad \text{iff} \quad \begin{array}{l} 0 \neq \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{and} \\ \text{for all } \langle l,t'\rangle \in (L \times T): \\ \text{if } R^{\leq}_{\mathrm{T}}\ \langle l,t\rangle\ \langle l,t'\rangle \\ \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

EVERYTIME BUT DIFFERENT (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t\rangle \models_{\mathrm{LTL}} \triangle^{\mathrm{T}}_{=\tau}\, \varphi \quad \text{iff} \quad \begin{array}{l} 0 \neq \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{and} \\ \text{for all } \langle l,t'\rangle \in (L \times T): \\ \text{if } \langle l,t\rangle \neq_{\mathrm{LT}} \langle l,t'\rangle \\ \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

EVERYTIME (AT A CERTAIN DISTANCE)

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t\rangle \models_{\mathrm{LTL}} \triangleq^{\mathrm{T}}_{=\tau}\, \varphi \quad \text{iff} \quad \begin{array}{l} \text{for all } \langle l,t'\rangle \in (L \times T): \\ \text{if } R^{\mathrm{U}}_{\mathrm{T}}\ \langle l,t'\rangle\ \langle l,t\rangle \\ \quad \text{and } \rho_{\mathrm{T}}(t,t') = \alpha_{\mathrm{T}}(\langle l,t\rangle, \tau) \\ \text{then } \mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t'\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

LOCATIVE QUANTIFICATION

$$\mathcal{M}^{||}_{\mathrm{LT}}, \langle l,t\rangle \models_{\mathrm{LTL}} \forall\zeta : \varphi \quad \text{iff} \quad \begin{array}{l} \text{for all } \alpha'_{\mathrm{L}} \text{ different} \\ \text{from } \alpha_{\mathrm{L}} \text{ just on } \zeta: \\ \mathcal{M}^{||}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}], \langle l,t\rangle \models_{\mathrm{LTL}} \varphi \end{array}$$

TEMPORAL QUANTIFICATION

$$\mathcal{M}^{|\cdot|}_{\mathrm{LT}}, \langle l, t\rangle \models_{\mathrm{LTL}} \forall\delta : \varphi \quad \text{iff} \quad \begin{array}{l}\text{for all bindings } \alpha'_{\mathrm{T}} \text{ different}\\ \text{from } \alpha_{\mathrm{T}} \text{ just on } \delta:\\ \mathcal{M}^{|\cdot|}_{\mathrm{LT}}[\alpha'_{\mathrm{T}}/\alpha_{\mathrm{T}}], \langle l, t\rangle \models_{\mathrm{LTL}} \varphi\end{array}$$

Thereby, $\mathcal{M}^{|\cdot|}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}]$ results from $\mathcal{M}^{|\cdot|}_{\mathrm{LT}}$ by replacing α_{L} with the new binding α'_{L}, i.e., $\mathcal{M}^{|\cdot|}_{\mathrm{LT}}[\alpha'_{\mathrm{L}}/\alpha_{\mathrm{L}}] = (\mathcal{F}^{|\cdot|}_{\mathrm{LT}}, \alpha'_{\mathrm{L}}, \alpha_{\mathrm{T}}, \mathcal{I}_{\mathrm{LT}})$. Substitution for the temporal binding is defined analogously.[2]

A.3 DEDUCTION

The deduction system of LTL comprises definitions of non-primitive operators and, of course, axioms and rules for locative temporal consequence. The version to be presented here has been used for referencing in part II of this thesis.

In the sequel, we assume that x is a variable temporal distance symbol, a variable locative distance symbol, or a variable symbol of some other sort that we have avoided to mention here explicitly (see beginning of section A.1).

A.3.1 DEFINITIONS

PROPOSITIONAL CALCULUS

(LTL-1) $\neg\varphi \stackrel{\text{def}}{=} \varphi \rightarrow \bot$ (Negation)

(LTL-2) $\top \stackrel{\text{def}}{=} \neg\bot$ (Truth)

(LTL-3) $\varphi_1 \wedge \varphi_2 \stackrel{\text{def}}{=} \neg(\varphi_1 \rightarrow \neg\varphi_2)$ (Conjunction)

(LTL-4) $\varphi_1 \vee \varphi_2 \stackrel{\text{def}}{=} \neg\varphi_1 \rightarrow \varphi_2$ (Disjunction)

(LTL-5) $\varphi_1 \leftrightarrow \varphi_2 \stackrel{\text{def}}{=} (\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)$ (Equivalence)

FIRST-ORDER PREDICATE CALCULUS

(LTL-6) $\exists x : \varphi \stackrel{\text{def}}{=} \neg\forall x : \neg\varphi$ (Existential Quantification)

(LTL-7) $\eta_1 \leq \eta_2 \stackrel{\text{def}}{=} \eta_1 < \eta_2 \vee \eta_1 = \eta_2$ (L-Less-Equal)

[2]Note that the universal relations $R^{\mathrm{U}}_{\mathrm{L}}$ and $R^{\mathrm{U}}_{\mathrm{T}}$ have been introduced for reasons of analogy between modal operators and relations on the corresponding universes.

(LTL-8) $\tau_1 \leq \tau_2 \stackrel{\text{def}}{=} \tau_1 < \tau_2 \vee \tau_1 = \tau_2$ (T-Less-Equal)

LOCATIVE CALCULUS (REFLEXIVE)

(LTL-9) $\boxdot^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \forall \zeta : \boxdot^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere in the Class)

(LTL-10) $\boxdot^{\mathrm{L}}_{<\eta} \varphi \stackrel{\text{def}}{=} \forall \zeta : \zeta < \eta \rightarrow \boxdot^{\mathrm{L}}_{=\zeta} \varphi$

(LTL-11) $\diamondsuit^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \exists \zeta : \diamondsuit^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere in the Class)

(LTL-12) $\diamondsuit^{\mathrm{L}}_{=\eta} \varphi \stackrel{\text{def}}{=} \neg \boxdot^{\mathrm{L}}_{=\eta} \neg \varphi$

(LTL-13) $\diamondsuit^{\mathrm{L}}_{<\eta} \varphi \stackrel{\text{def}}{=} \exists \zeta : \zeta < \eta \wedge \diamondsuit^{\mathrm{L}}_{=\zeta} \varphi$

(LTL-14) $\triangle^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \forall \zeta : 0 < \zeta \rightarrow \triangle^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere Else)

(LTL-15) $\triangledown^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \exists \zeta : 0 < \zeta \wedge \triangledown^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere Else)

(LTL-16) $\triangledown^{\mathrm{L}}_{=\eta} \varphi \stackrel{\text{def}}{=} \neg \triangle^{\mathrm{L}}_{=\eta} \neg \varphi$

(LTL-17) $\triangledown^{\mathrm{L}}_{<\eta} \varphi \stackrel{\text{def}}{=} \exists \zeta : 0 < \zeta < \eta \wedge \triangledown^{\mathrm{L}}_{=\zeta} \varphi$

(LTL-18) $\circledcirc^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \forall \zeta : \circledcirc^{\mathrm{L}}_{=\zeta} \varphi$ (Everywhere)

(LTL-19) $\circledtriangledown^{\mathrm{L}} \varphi \stackrel{\text{def}}{=} \exists \zeta : \circledtriangledown^{\mathrm{L}}_{=\zeta} \varphi$ (Somewhere)

(LTL-20) $\circledtriangledown^{\mathrm{L}}_{=\eta} \varphi \stackrel{\text{def}}{=} \neg \circledcirc^{\mathrm{L}}_{=\eta} \neg \varphi$

(LTL-21) $\circledtriangledown^{\mathrm{L}}_{<\eta} \varphi \stackrel{\text{def}}{=} \exists \zeta : 0 < \zeta < \eta \wedge \circledtriangledown^{\mathrm{L}}_{=\zeta} \varphi$

(LTL-22) $\circledtriangledown^{\mathrm{L}}_{\leq\eta} \varphi \stackrel{\text{def}}{=} \circledtriangledown^{\mathrm{L}}_{=\eta} \varphi \vee \circledtriangledown^{\mathrm{L}}_{<\eta} \varphi$

TEMPORAL CALCULUS (IRREFLEXIVE)

(LTL-23) $\boxminus^{\mathrm{T}} \varphi \stackrel{\text{def}}{=} \forall \delta : 0 < \delta \rightarrow \boxminus^{\mathrm{T}}_{=\delta} \varphi$ (Always in the Past)

(LTL-24) $\boxminus^{\mathrm{T}}_{<\tau} \varphi \stackrel{\text{def}}{=} \forall \delta : 0 < \delta < \tau \rightarrow \boxminus^{\mathrm{T}}_{=\delta} \varphi$

(LTL-25) $\diamondminus^{\mathrm{T}} \varphi \stackrel{\text{def}}{=} \exists \delta : 0 < \delta \wedge \diamondminus^{\mathrm{T}}_{=\delta} \varphi$ (Eventually in the Past)

(LTL-26) $\diamondminus^{\mathrm{T}}_{=\tau} \varphi \stackrel{\text{def}}{=} \neg \boxminus^{\mathrm{T}}_{=\tau} \neg \varphi$

(LTL-27) $\diamondminus^{\mathrm{T}}_{<\tau} \varphi \stackrel{\text{def}}{=} \exists \delta : 0 < \delta < \tau \wedge \diamondminus^{\mathrm{T}}_{=\delta} \varphi$

(LTL-28) $\Diamond^{\mathrm{T}}_{\leq\tau}\,\varphi \stackrel{\text{def}}{=} \Diamond^{\mathrm{T}}_{=\tau}\,\varphi \vee \Diamond^{\mathrm{T}}_{<\tau}\,\varphi$

(LTL-29) $\Diamond^{\mathrm{T}}_{>\tau}\,\varphi \stackrel{\text{def}}{=} \exists\tau' : \tau < \tau' \wedge \Diamond^{\mathrm{T}}_{=\tau'}\,\varphi$

(LTL-30) $\boxplus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta \rightarrow \boxplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Future)

(LTL-31) $\boxplus^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta < \tau \rightarrow \boxplus^{\mathrm{T}}_{=\delta}\,\varphi$

(LTL-32) $\oplus^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \oplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Future)

(LTL-33) $\oplus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\boxplus^{\mathrm{T}}_{=\tau}\,\neg\varphi$

(LTL-34) $\oplus^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta < \tau \wedge \oplus^{\mathrm{T}}_{=\delta}\,\varphi$

(LTL-35) $\oplus^{\mathrm{T}}_{\leq\tau}\,\varphi \stackrel{\text{def}}{=} \oplus^{\mathrm{T}}_{=\tau}\,\varphi \vee \oplus^{\mathrm{T}}_{<\tau}\,\varphi$

(LTL-36) $\Delta^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : 0 < \delta \rightarrow \Delta^{\mathrm{T}}_{=\delta}\,\varphi$ (Everytime but Different)

(LTL-37) $\nabla^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \nabla^{\mathrm{T}}_{=\delta}\,\varphi$ (Sometime but Different)

(LTL-38) $\nabla^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} \neg\Delta^{\mathrm{T}}_{=\tau}\,\neg\varphi$

(LTL-39) $\nabla^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\text{def}}{=} \exists\delta : 0 < \delta < \tau \wedge \nabla^{\mathrm{T}}_{=\delta}\,\varphi$

(LTL-40) $\varphi_1\ \mathrm{S}\ \varphi_2 \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \varphi_1\ \mathrm{S}_{=\delta}\ \varphi_2$ (Since)

(LTL-41) $\varphi_1\ \mathrm{S}_{=\tau}\ \varphi_2 \stackrel{\text{def}}{=} \Diamond^{\mathrm{T}}_{=\tau}\,\varphi_2 \wedge \boxminus^{\mathrm{T}}_{<\tau}\,\varphi_1$

(LTL-42) $\varphi_1\ \mathrm{U}\ \varphi_2 \stackrel{\text{def}}{=} \exists\delta : 0 < \delta \wedge \varphi_1\ \mathrm{U}_{=\delta}\ \varphi_2$ (Until)

(LTL-43) $\varphi_1\ \mathrm{U}_{=\tau}\ \varphi_2 \stackrel{\text{def}}{=} \oplus^{\mathrm{T}}_{=\tau}\,\varphi_2 \wedge \boxplus^{\mathrm{T}}_{<\tau}\,\varphi_1$

TEMPORAL CALCULUS (REFLEXIVE)

(LTL-44) $\circledcirc^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \forall\delta : \circledcirc^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Past)

(LTL-45) $\circledcirc^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\text{def}}{=} (\tau = 0 \wedge \varphi) \vee \boxminus^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-46) $\circledcirc^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\text{def}}{=} \forall\tau' : \tau' < \tau \rightarrow \circledcirc^{\mathrm{T}}_{=\tau'}\,\varphi$

(LTL-47) $\circledcirc^{\mathrm{T}}_{\leq\tau}\,\varphi \stackrel{\text{def}}{=} \circledcirc^{\mathrm{T}}_{=\tau}\,\varphi \wedge \circledcirc^{\mathrm{T}}_{<\tau}\,\varphi$

(LTL-48) $\circledast^{\mathrm{T}}\varphi \stackrel{\text{def}}{=} \exists\delta : \circledast^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Past)

(LTL-49) $\diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\mathrm{def}}{=} (\tau = 0 \wedge \varphi) \vee \diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-50) $\boxplus^{\mathrm{T}}\varphi \stackrel{\mathrm{def}}{=} \forall\delta : \boxplus^{\mathrm{T}}_{=\delta}\,\varphi$ (Always in the Future)

(LTL-51) $\boxplus^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\mathrm{def}}{=} (\tau = 0 \wedge \varphi) \vee \boxplus^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-52) $\boxplus^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\mathrm{def}}{=} \forall\tau' : \tau' < \tau \rightarrow \boxplus^{\mathrm{T}}_{=\tau'}\,\varphi$

(LTL-53) $\boxplus^{\mathrm{T}}_{\leq\tau}\,\varphi \stackrel{\mathrm{def}}{=} \boxplus^{\mathrm{T}}_{=\tau}\,\varphi \wedge \boxplus^{\mathrm{T}}_{<\tau}\,\varphi$

(LTL-54) $\diamondsuit^{\mathrm{T}}\varphi \stackrel{\mathrm{def}}{=} \exists\delta : \diamondsuit^{\mathrm{T}}_{=\delta}\,\varphi$ (Eventually in the Future)

(LTL-55) $\diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\mathrm{def}}{=} (\tau = 0 \wedge \varphi) \vee \diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-56) $\vartriangle^{\mathrm{T}}\,\varphi \stackrel{\mathrm{def}}{=} \forall\delta : \vartriangle^{\mathrm{T}}_{=\delta}\,\varphi$ (Everytime)

(LTL-57) $\triangledown^{\mathrm{T}}\varphi \stackrel{\mathrm{def}}{=} \exists\delta : \triangledown^{\mathrm{T}}_{=\delta}\,\varphi$ (Sometime)

(LTL-58) $\triangledown^{\mathrm{T}}_{=\tau}\,\varphi \stackrel{\mathrm{def}}{=} \neg\vartriangle^{\mathrm{T}}_{=\tau}\,\neg\varphi$

(LTL-59) $\triangledown^{\mathrm{T}}_{<\tau}\,\varphi \stackrel{\mathrm{def}}{=} \exists\delta : \delta < \tau \wedge \triangledown^{\mathrm{T}}_{=\delta}\,\varphi$

LOCATIVE TEMPORAL CALCULUS (IRREFLEXIVE TIME)

(LTL-60) $\boxminus^{\mathrm{LT}}\varphi \stackrel{\mathrm{def}}{=} \boxdot^{\mathrm{L}}\boxminus^{\mathrm{T}}\varphi$ (Everywhere in the Class/Always in the Past)

(LTL-61) $\boxminus^{\mathrm{LT}}_{=(\eta,\tau)}\,\varphi \stackrel{\mathrm{def}}{=} \boxdot^{\mathrm{L}}_{=\eta}\,\boxminus^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-62) $\boxplus^{\mathrm{LT}}\varphi \stackrel{\mathrm{def}}{=} \boxdot^{\mathrm{L}}\boxplus^{\mathrm{T}}\varphi$ (Everywhere in the Class/Always in the Future)

(LTL-63) $\boxplus^{\mathrm{LT}}_{=(\eta,\tau)}\,\varphi \stackrel{\mathrm{def}}{=} \boxdot^{\mathrm{L}}_{=\eta}\,\boxplus^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-64) $\diamondsuit^{\mathrm{LT}}\varphi \stackrel{\mathrm{def}}{=} \diamondsuit^{\mathrm{L}}\diamondsuit^{\mathrm{T}}\varphi$ (Somewhere in the Class/Eventually in the Past)

(LTL-65) $\diamondsuit^{\mathrm{LT}}_{=(\eta,\tau)}\,\varphi \stackrel{\mathrm{def}}{=} \diamondsuit^{\mathrm{L}}_{=\eta}\,\diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-66) $\diamondsuit^{\mathrm{LT}}\varphi \stackrel{\mathrm{def}}{=} \diamondsuit^{\mathrm{L}}\diamondsuit^{\mathrm{T}}\varphi$ (Somewhere in the Class/ Eventually in the Future)

(LTL-67) $\diamondsuit^{\mathrm{LT}}_{=(\eta,\tau)}\,\varphi \stackrel{\mathrm{def}}{=} \diamondsuit^{\mathrm{L}}_{=\eta}\,\diamondsuit^{\mathrm{T}}_{=\tau}\,\varphi$

(LTL-68) $\triangledown^{\mathrm{LT}}\varphi \stackrel{\mathrm{def}}{=} \triangledown^{\mathrm{L}}\triangledown^{\mathrm{T}}\varphi$ (Somewhere Sometime but Different)

LOCATIVE TEMPORAL CALCULUS (REFLEXIVE TIME)

(LTL-69) $\boxminus^{LT}\varphi \stackrel{def}{=} \boxdot^{L}\boxminus^{T}\varphi$ (Everywhere in the Class/Always in the Past)

(LTL-70) $\boxminus^{LT}_{=(\eta,\tau)}\ \varphi \stackrel{def}{=} \boxdot^{L}_{=\eta}\ \boxminus^{T}_{=\tau}\ \varphi$

(LTL-71) $\boxplus^{LT}\varphi \stackrel{def}{=} \boxdot^{L}\boxplus^{T}\varphi$ (Everywhere in the Class/Always in the Future)

(LTL-72) $\boxplus^{LT}_{=(\eta,\tau)}\ \varphi \stackrel{def}{=} \boxdot^{L}_{=\eta}\ \boxplus^{T}_{=\tau}\ \varphi$

(LTL-73) $\diamondminus^{LT}\varphi \stackrel{def}{=} \diamond^{L}\diamondminus^{T}\varphi$ (Somewhere in the Class/Eventually in the Past)

(LTL-74) $\diamondminus^{LT}_{=(\eta,\tau)}\ \varphi \stackrel{def}{=} \diamond^{L}_{=\eta}\ \diamondminus^{T}_{=\tau}\ \varphi$

(LTL-75) $\diamondplus^{LT}\varphi \stackrel{def}{=} \diamond^{L}\diamondplus^{T}\varphi$ (Somewhere in the Class/ Eventually in the Future)

(LTL-76) $\diamondplus^{LT}_{=(\eta,\tau)}\ \varphi \stackrel{def}{=} \diamond^{L}_{=\eta}\ \diamondplus^{T}_{=\tau}\ \varphi$

METRIC CALCULUS

(LTL-77) $(\eta \oplus 1) \otimes \tau \stackrel{def}{=} (\eta \otimes \tau) \oplus \tau$ (Multiplication)

(LTL-78) $0 \otimes \tau \stackrel{def}{=} 0$

A.3.2 AXIOMS

PROPOSITIONAL CALCULUS

(LTL-79) $\vdash_{LTL} (\varphi_1 \rightarrow (\varphi_2 \rightarrow \varphi_3)) \rightarrow ((\varphi_1 \rightarrow \varphi_2) \rightarrow (\varphi_1 \rightarrow \varphi_3))$ (Frege's Syllogism)

(LTL-80) $\vdash_{LTL} \varphi_1 \rightarrow (\varphi_2 \rightarrow \varphi_1)$ (Monotonicity)

(LTL-81) $\vdash_{LTL} (\neg\varphi_2 \rightarrow \neg\varphi_1) \rightarrow (\varphi_1 \rightarrow \varphi_2)$ (Transposition)

FIRST-ORDER PREDICATE CALCULUS

(LTL-82) $\vdash_{LTL} \forall x : (\varphi_1 \rightarrow \varphi_2) \rightarrow (\varphi_1 \rightarrow (\forall x : \varphi_2))$ (Quantifier Distributivity)
Assume that x does not occur free in φ_1.

(LTL-83) $\vdash_{LTL} (\forall x : \varphi_1) \rightarrow \varphi_2$ (Quantifier Elimination)
Assume that φ_2 is obtained from φ_1 by substituting all occurrences of x by a well-formed term of the corresponding sort.

(LTL-84) $\vdash_{\mathrm{LTL}} (\forall x : \boxdot^{\mathrm{L}} \varphi) \to \boxdot^{\mathrm{L}} (\forall x : \varphi)$ (Barcan Formula)

(LTL-85) $\vdash_{\mathrm{LTL}} (\forall x : \boxplus^{\mathrm{T}} \varphi) \to \boxplus^{\mathrm{T}} \forall x : \varphi$

LOCATIVE CALCULUS

(LTL-86) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}} p \to p$ (L-Reflexivity)

(LTL-87) $\vdash_{\mathrm{LTL}} p \to \boxdot^{\mathrm{L}} \diamondsuit^{\mathrm{L}} p$ (L-Symmetry)

(LTL-88) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}} p \to \boxdot^{\mathrm{L}} \boxdot^{\mathrm{L}} p$ (L-Transitivity)

LOCATIVE METRIC CALCULUS

(LTL-89) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = 0 \to \zeta_1 = 0 \wedge \zeta_2 = 0$ ($\mathbb{ID}_{\mathrm{L}}^{0,+}$)

(LTL-90) $\vdash_{\mathrm{LTL}} \forall \zeta : \zeta \oplus 0 = \zeta$ ($\oplus$-Identity)

(LTL-91) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2, \zeta_3 : \zeta_1 \oplus \zeta_2 = \zeta_1 \oplus \zeta_3 \to \zeta_2 = \zeta_3$ ($\oplus$-Left Injectivity)

(LTL-92) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2, \zeta_3 : \zeta_1 \oplus \zeta_3 = \zeta_2 \oplus \zeta_3 \to \zeta_1 = \zeta_2$ ($\oplus$-Right Injectivity)

(LTL-93) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2, \zeta_3 : (\zeta_1 \oplus \zeta_2) \oplus \zeta_3 = \zeta_1 \oplus (\zeta_2 \oplus \zeta_3)$ ($\oplus$-Associativity)

(LTL-94) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2 : \zeta_1 \oplus \zeta_2 = \zeta_2 \oplus \zeta_1$ ($\oplus$-Commutativity)

(LTL-95) $\vdash_{\mathrm{LTL}} \forall \zeta : \mathbb{V}^{\mathrm{L}} \mathbb{V}^{\mathrm{L}}_{=\zeta} \top$ ($\mathbb{ID}_{\mathrm{L}}$-Surjectivity)

(LTL-96) $\vdash_{\mathrm{LTL}} p \leftrightarrow \mathbb{V}^{\mathrm{L}}_{=0} p$ (Zero L-Distance)

(LTL-97) $\vdash_{\mathrm{LTL}} \forall \zeta : [\, (p \wedge \mathbb{V}^{\mathrm{L}}_{=\zeta} q) \to \mathbb{V}^{\mathrm{L}}_{=\zeta} (q \wedge \mathbb{V}^{\mathrm{L}}_{=\zeta} p) \,]$ (Symmetric L-Distance)

(LTL-98) $\vdash_{\mathrm{LTL}} \forall \zeta_1, \zeta_2 : [\, \diamondsuit^{\mathrm{L}}_{=\zeta_1} \diamondsuit^{\mathrm{L}}_{=\zeta_2} p \to \mathbb{V}^{\mathrm{L}}_{\leq \zeta_1 + \zeta_2} p \,]$
(Conditional L-Inequality)

(LTL-99) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} (\varphi_1 \to \varphi_2) \to (\boxdot^{\mathrm{L}}_{=\eta} \varphi_1 \to \boxdot^{\mathrm{L}}_{=\eta} \varphi_2)$ (L-Distributivity)

(LTL-100) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \boxdot^{\mathrm{L}}_{=1} \varphi \leftrightarrow \boxdot^{\mathrm{L}}_{=\eta \oplus 1} \varphi$ (L-Addition)

(LTL-101) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}} \boxdot^{\mathrm{L}}_{=\eta} \varphi \leftrightarrow \boxdot^{\mathrm{L}}_{=\eta} \varphi$ (L-Finiteness)

TEMPORAL CALCULUS

(LTL-102) $\vdash_{\mathrm{LTL}} \diamondplus^{\mathrm{T}} p \to \triangledown^{\mathrm{T}} p$ (T-Irreflexivity)

(LTL-103) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}} \oplus^{\mathrm{T}} p \rightarrow \oplus^{\mathrm{T}} p$ (T-Transitivity)

(LTL-104) $\vdash_{\mathrm{LTL}} \nabla^{\mathrm{T}} p \rightarrow (\ominus^{\mathrm{T}} p \vee \oplus^{\mathrm{T}} p)$ (T-Connectedness)

(LTL-105) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}} p \rightarrow \oplus^{\mathrm{T}} \oplus^{\mathrm{T}} p$ (T-Denseness)

(LTL-106) $\vdash_{\mathrm{LTL}} \boxplus^{\mathrm{T}} p \rightarrow \oplus^{\mathrm{T}} p$ (No T-End)

TEMPORAL METRIC CALCULUS

(LTL-107) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2 : \delta_1 \oplus \delta_2 = 0 \rightarrow \delta_1 = 0 \wedge \delta_2 = 0$ $(\mathbb{ID}_{\mathrm{T}}^{0,+})$

(LTL-108) $\vdash_{\mathrm{LTL}} \forall \delta : \delta \oplus 0 = \delta$ ($\oplus$-Identity)

(LTL-109) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2, \delta_3 : \delta_1 \oplus \delta_2 = \delta_1 \oplus \delta_3 \rightarrow \delta_2 = \delta_3$ ($\oplus$-Left Injectivity)

(LTL-110) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2, \delta_3 : \delta_1 \oplus \delta_3 = \delta_2 \oplus \delta_3 \rightarrow \delta_1 = \delta_2$ ($\oplus$-Right Injectivity)

(LTL-111) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2, \delta_3 : (\delta_1 \oplus \delta_2) \oplus \delta_3 = \delta_1 \oplus (\delta_2 \oplus \delta_3)$ ($\oplus$-Associativity)

(LTL-112) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2 : \delta_1 \oplus \delta_2 = \delta_2 \oplus \delta_1$ ($\oplus$-Commutativity)

(LTL-113) $\vdash_{\mathrm{LTL}} \forall \delta : \mathbb{V}^{\mathrm{T}} \mathbb{V}^{\mathrm{T}}_{=\delta} \top$ (T-Surjectivity)

(LTL-114) $\vdash_{\mathrm{LTL}} p \leftrightarrow \mathbb{V}^{\mathrm{T}}_{=0} p$ (Zero T-Distance)

(LTL-115) $\vdash_{\mathrm{LTL}} \forall \delta : [\ (p \wedge \mathbb{V}^{\mathrm{T}}_{=\delta} q) \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta} (q \wedge \mathbb{V}^{\mathrm{L}}_{=\delta} p)\ ]$
(Symmetric T-Distance)

(LTL-116) $\vdash_{\mathrm{LTL}} \forall \delta_1, \delta_2 : [\ (\oplus^{\mathrm{T}}_{=\delta_1} \oplus^{\mathrm{T}}_{=\delta_2} p \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta_1+\delta_2} p)$
$\wedge$ (Conditional T-Equality)
$(\ominus^{\mathrm{T}}_{=\delta_1} \ominus^{\mathrm{T}}_{=\delta_2} p \rightarrow \mathbb{V}^{\mathrm{T}}_{=\delta_1+\delta_2} p)$
$]$

(LTL-117) $\vdash_{\mathrm{LTL}} \boxplus^{\mathrm{T}}_{=\tau} (\varphi_1 \rightarrow \varphi_2) \rightarrow (\boxplus^{\mathrm{T}}_{=\tau} \varphi_1 \rightarrow \boxplus^{\mathrm{T}}_{=\tau} \varphi_2)$ (FT-Distributivity)

(LTL-118) $\vdash_{\mathrm{LTL}} \boxminus^{\mathrm{T}}_{=\tau} (\varphi_1 \rightarrow \varphi_2) \rightarrow (\boxminus^{\mathrm{T}}_{=\tau} \varphi_1 \rightarrow \boxminus^{\mathrm{T}}_{=\tau} \varphi_2)$ (PT-Distributivity)

(LTL-119) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=\tau} \varphi \leftrightarrow \oplus^{\mathrm{T}}_{=\tau} \top \wedge \boxplus^{\mathrm{T}}_{=\tau} \varphi$

(LTL-120) $\vdash_{\mathrm{LTL}} \ominus^{\mathrm{T}}_{=\tau} \varphi \leftrightarrow \ominus^{\mathrm{T}}_{=\tau} \top \wedge \boxminus^{\mathrm{T}}_{=\tau} \varphi$

(LTL-121) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=0} \varphi \leftrightarrow \bot \leftrightarrow \ominus^{\mathrm{T}}_{=0} \varphi$

(LTL-122) $\vdash_{\mathrm{LTL}} \ominus^{\mathrm{T}}_{=\tau_1} \oplus^{\mathrm{T}}_{=\tau_2} \varphi \leftrightarrow \ominus^{\mathrm{T}}_{=\tau_1} \top \wedge \ominus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \varphi$ (Addition to Past)

(LTL-123) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=\tau_1} \oplus^{\mathrm{T}}_{=\tau_2} \varphi \leftrightarrow \oplus^{\mathrm{T}}_{=\tau_1} \top \wedge \oplus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \varphi$ (Addition to Future)

(LTL-124) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=\tau_1} \ominus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \varphi \leftrightarrow \oplus^{\mathrm{T}}_{=\tau_1} \top \wedge \ominus^{\mathrm{T}}_{=\tau_2} \varphi$ (FP-Mixing)

(LTL-125) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \ominus^{\mathrm{T}}_{=\tau_1} \varphi \leftrightarrow \oplus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \top \wedge \oplus^{\mathrm{T}}_{=\tau_2} \varphi$

(LTL-126) $\vdash_{\mathrm{LTL}} \ominus^{\mathrm{T}}_{=\tau_1} \oplus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \varphi \leftrightarrow \ominus^{\mathrm{T}}_{=\tau_1} \top \wedge \oplus^{\mathrm{T}}_{=\tau_2} \varphi$

(LTL-127) $\vdash_{\mathrm{LTL}} \ominus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \oplus^{\mathrm{T}}_{=\tau_1} \varphi \leftrightarrow \ominus^{\mathrm{T}}_{=\tau_1 \oplus \tau_2} \top \wedge \ominus^{\mathrm{T}}_{=\tau_2} \varphi$

(LTL-128) $\vdash_{\mathrm{LTL}} \forall \delta : [\, 0 < \delta \rightarrow \oplus^{\mathrm{T}}_{=\delta} \top \,]$ (Local T-Surjectivity)

LOCATIVE TEMPORAL METRIC CALCULUS

(LTL-129) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \boxplus^{\mathrm{T}}_{=\tau} \varphi \leftrightarrow \boxplus^{\mathrm{T}}_{=\tau} \boxdot^{\mathrm{L}}_{=\eta} \varphi$ (LT-Confluency)

(LTL-130) $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \boxminus^{\mathrm{T}}_{=\tau} \varphi \leftrightarrow \boxminus^{\mathrm{T}}_{=\tau} \boxdot^{\mathrm{L}}_{=\eta} \varphi$

(LTL-131) $\vdash_{\mathrm{LTL}} \triangle^{\mathrm{L}}_{=\eta} \triangle^{\mathrm{T}}_{=\tau} \varphi \leftrightarrow \triangle^{\mathrm{T}}_{=\tau} \triangle^{\mathrm{L}}_{=\eta} \varphi$

(LTL-132) $\vdash_{\mathrm{LTL}} \diamondsuit^{\mathrm{L}}_{=\eta} \boxplus^{\mathrm{T}}_{=\tau} \varphi \rightarrow \boxplus^{\mathrm{T}}_{=\tau} \diamondsuit^{\mathrm{L}}_{=\eta} \varphi$ (LT-Commutativity)

(LTL-133) $\vdash_{\mathrm{LTL}} \diamondsuit^{\mathrm{L}}_{=\eta} \boxminus^{\mathrm{T}}_{=\tau} \varphi \rightarrow \boxminus^{\mathrm{T}}_{=\tau} \diamondsuit^{\mathrm{L}}_{=\eta} \varphi$

(LTL-134) $\vdash_{\mathrm{LTL}} \oplus^{\mathrm{T}}_{=\tau} \boxdot^{\mathrm{L}}_{=\eta} \varphi \rightarrow \boxdot^{\mathrm{L}}_{=\eta} \oplus^{\mathrm{T}}_{=\tau} \varphi$ (TL-Commutativity)

(LTL-135) $\vdash_{\mathrm{LTL}} \ominus^{\mathrm{T}}_{=\tau} \boxdot^{\mathrm{L}}_{=\eta} \varphi \rightarrow \boxdot^{\mathrm{L}}_{=\eta} \ominus^{\mathrm{T}}_{=\tau} \varphi$

A.3.3 RULES

(LTL-136) whenever $\vdash_{\mathrm{LTL}} \varphi_1 \rightarrow \varphi_2$ and $\vdash_{\mathrm{LTL}} \varphi_1$ then $\vdash_{\mathrm{LTL}} \varphi_2$ (Modus Ponens)

(LTL-137) whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \forall x : \varphi$ (Generalization)

(LTL-138) whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \boxdot^{\mathrm{L}}_{=\eta} \varphi$ (L-Necessitation)

(LTL-139) whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \boxplus^{\mathrm{T}}_{=\tau} \varphi$ (FT-Necessitation)

(LTL-140) whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \boxminus^{\mathrm{T}}_{=\tau} \varphi$ (PT-Necessitation)

A.3.4 SOME PROPERTIES

Properties for the metric temporal calculus have already been stated by Koymans [54]. Some of those properties will be adopted to our metric locative temporal calculus below.

LOCAL DYNAMIC PROPERTIES

Recall from chapter 4 that LTL-formulae denoting *local dynamic* properties only contain locative temporal connectives with a superscript T such that the locative reference point will be fixed during evaluation of these formulae.

The following proposition can be derived mainly from axiom (LTL-124) by setting $\tau_1 = \tau$ and $\tau_2 = 0$.

PROPOSITION A.1 Let $0 < \tau$.

$$\oplus^{\mathsf{T}}_{=\tau} \ominus^{\mathsf{T}}_{=\tau} \varphi \leftrightarrow \varphi \wedge \oplus^{\mathsf{T}}_{=\tau} \top$$

$\Diamond$

PROPOSITION A.2 Let $0 < \tau$.

$$\oplus^{\mathsf{T}}_{\leq\tau} \oplus^{\mathsf{T}}_{=\tau} \varphi \rightarrow \oplus^{\mathsf{T}}_{\leq\tau} \varphi$$

$\Diamond$

PROPOSITION A.3 Let $0 < \tau$.

$$\oplus^{\mathsf{T}}_{\leq\tau} \ominus^{\mathsf{T}}_{\leq\tau} \varphi \rightarrow \oplus^{\mathsf{T}}_{\leq\tau} \varphi$$

$\Diamond$

PROPOSITION A.4 Let $0 < \tau$.

$$\oplus^{\mathsf{T}}_{=\tau} \boxminus^{\mathsf{T}} \varphi \rightarrow \boxminus^{\mathsf{T}}_{\geq\tau} \varphi$$

$\Diamond$

PROPOSITION A.5

$$\varphi \wedge \boxminus^{\mathsf{T}} \varphi \rightarrow \boxminus^{\mathsf{T}} \varphi$$

$\Diamond$

PROPOSITION A.6

$$\exists \delta : [\, \delta > 0 \wedge \boxminus^{\mathsf{T}}_{\geq\delta} \varphi \wedge \boxminus^{\mathsf{T}}_{<\delta} \varphi \,] \rightarrow \boxminus^{\mathsf{T}} \varphi$$

$\Diamond$

A proof of the following two propositions, i.e., propositions A.7 and A.8 will proceed analogously to the corresponding proofs in the metric temporal calculus (cf. Koymans [54], page 126–128).

PROPOSITION A.7

$$\boxplus^{\mathrm{T}}_{=\tau}\ (\varphi_1 \wedge \varphi_2) \leftrightarrow (\boxplus^{\mathrm{T}}_{=\tau}\ \varphi_1 \wedge \boxplus^{\mathrm{T}}_{=\tau}\ \varphi_2)$$

◇

PROPOSITION A.8

$$\diamondplus^{\mathrm{T}}_{=\tau}\ (\varphi_1 \wedge \varphi_2) \leftrightarrow (\diamondplus^{\mathrm{T}}_{=\tau}\ \varphi_1 \wedge \diamondplus^{\mathrm{T}}_{=\tau}\ \varphi_2)$$

◇

Instead of an =-symbol in the subscripts of the previous two propositions we can also place a <-symbol (see propositions A.9 and A.10). But note that only in the first case the old equivalence will be maintained whereas in the second case we will merely get an implication.

PROPOSITION A.9

$$\boxplus^{\mathrm{T}}_{<\tau}\ (\varphi_1 \wedge \varphi_2) \leftrightarrow (\boxplus^{\mathrm{T}}_{<\tau}\ \varphi_1 \wedge \boxplus^{\mathrm{T}}_{<\tau}\ \varphi_2)$$

◇

PROPOSITION A.10

$$\diamondplus^{\mathrm{T}}_{<\tau}\ (\varphi_1 \wedge \varphi_2) \rightarrow (\diamondplus^{\mathrm{T}}_{<\tau}\ \varphi_1 \wedge \diamondplus^{\mathrm{T}}_{<\tau}\ \varphi_2)$$

◇

From proposition A.8 and A.10 we can then derive the combination of "less-than" and "equal":

PROPOSITION A.11

$$\diamondplus^{\mathrm{T}}_{\leq\tau}\ (\varphi_1 \wedge \varphi_2) \rightarrow (\diamondplus^{\mathrm{T}}_{\leq\tau}\ \varphi_1 \wedge \diamondplus^{\mathrm{T}}_{\leq\tau}\ \varphi_2)$$

◇

PROPOSITION A.12

$$\circledplus^{\mathrm{T}}_{\leq\tau}\ (\varphi_1 \rightarrow \varphi_2) \rightarrow (\circledplus^{\mathrm{T}}_{\leq\tau}\ \varphi_1 \rightarrow \circledplus^{\mathrm{T}}_{\leq\tau}\ \varphi_2)$$

◇

PROPOSITION A.13

$$\oplus^{\mathrm{T}}_{\leq\tau}\ (\varphi_1 \rightarrow \varphi_2) \rightarrow (\oplus^{\mathrm{T}}_{\leq\tau}\ \varphi_1 \rightarrow \oplus^{\mathrm{T}}_{\leq\tau}\ \varphi_2)$$

◊

From linearity we can derive the following proposition (cf. also Koymans [54]):

PROPOSITION A.14

$$\nabla^{\mathrm{T}}\varphi \leftrightarrow \diamondsuit^{\mathrm{T}}\varphi \vee \oplus^{\mathrm{T}}\varphi$$

◊

PROPOSITION A.15

$$\neg\nabla^{\mathrm{T}}\neg\varphi \leftrightarrow \Delta^{\mathrm{T}}\ \varphi$$

◊

PROPOSITION A.16 Let $0 < \tau$.

$$\Delta^{\mathrm{T}}\ \varphi \rightarrow \oplus^{\mathrm{T}}_{\leq\tau}\ \varphi$$

◊

The following proposition is easily derived from axiom (LTL-119) and axiom (LTL-128) (cf. also Koymans [54]).

PROPOSITION A.17

$$\oplus^{\mathrm{T}}_{=\tau}\ \varphi \leftrightarrow \boxplus^{\mathrm{T}}_{=\tau}\ \varphi$$

◊

PROPOSITION A.18

$$\circledast^{\mathrm{T}}_{=\tau}\ \varphi \leftrightarrow \boxplus^{\mathrm{T}}_{=\tau}\ \varphi$$

◊

PROPOSITION A.19

$$\neg\circledast^{\mathrm{T}}\varphi \leftrightarrow \boxminus^{\mathrm{T}}\neg\varphi$$

◊

PROPOSITION A.20

$$\neg\circledast^{\mathrm{T}}\varphi \leftrightarrow \boxplus^{\mathrm{T}}\neg\varphi$$

◊

PROPOSITION A.21

$$\neg \nabla^{\mathrm{T}} \varphi \leftrightarrow \Delta^{\mathrm{T}} \neg \varphi$$

◊

PROPOSITION A.22 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\boxplus^{\mathrm{T}}_{\leq \tau_1} \varphi \wedge \oplus^{\mathrm{T}}_{=\tau_1} \boxplus^{\mathrm{T}}_{\leq \tau_2} \varphi \leftrightarrow \boxplus^{\mathrm{T}}_{\leq \tau_1 \oplus \tau_2} \varphi$$

◊

PROPOSITION A.23 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\boxplus^{\mathrm{T}}_{\leq \tau_1 \oplus \tau_2} \varphi \rightarrow \boxplus^{\mathrm{T}}_{\leq \tau_1} \varphi \wedge \oplus^{\mathrm{T}}_{\leq \tau_1} \boxplus^{\mathrm{T}}_{\leq \tau_2} \varphi$$

◊

PROPOSITION A.24 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\boxplus^{\mathrm{T}}_{=\tau_1} \oplus^{\mathrm{T}}_{\leq \tau_2} \varphi \rightarrow \oplus^{\mathrm{T}}_{\leq \tau_1 \oplus \tau_2} \varphi$$

◊

PROPOSITION A.25 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\oplus^{\mathrm{T}}_{\leq \tau_1} \varphi \wedge \tau_1 \leq \tau_2 \rightarrow \oplus^{\mathrm{T}}_{\leq \tau_2} \varphi$$

◊

PROPOSITION A.26 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\boxplus^{\mathrm{T}}_{\leq \tau_1} \varphi \wedge \tau_2 \leq \tau_1 \rightarrow \boxplus^{\mathrm{T}}_{\leq \tau_2} \varphi$$

◊

PROPOSITION A.27 Let $0 < \tau_1$ and $0 < \tau_2$.

$$\oplus^{\mathrm{T}}_{\leq \tau_1} \oplus^{\mathrm{T}}_{\leq \tau_2} \varphi \rightarrow \oplus^{\mathrm{T}}_{\leq \tau_1 \oplus \tau_2} \varphi$$

◊

PROPOSITION A.28 Let $0 < \tau$.

$$\oplus^{\mathrm{T}}_{=\tau} \varphi \rightarrow \oplus^{\mathrm{T}}_{\leq \tau} \varphi$$

◊

PROPOSITION A.29

$$\oplus^{\mathrm{T}}_{=\tau}\,\varphi \rightarrow \oplus^{\mathrm{T}}_{\leq\tau}\,\varphi$$

$\Diamond$

PROPOSITION A.30

$$\oplus^{\mathrm{T}}_{<\tau}\,\varphi \rightarrow \oplus^{\mathrm{T}}_{\leq\tau}\,\varphi$$

$\Diamond$

PROPOSITION A.31

$$\oplus^{\mathrm{T}}\oplus^{\mathrm{T}}\varphi \rightarrow \oplus^{\mathrm{T}}\varphi$$

$\Diamond$

PROPOSITION A.32

$$\ominus^{\mathrm{T}}\ominus^{\mathrm{T}}\varphi \rightarrow \ominus^{\mathrm{T}}\varphi$$

$\Diamond$

PROPOSITION A.33

$$\ominus^{\mathrm{T}}\triangle^{\mathrm{T}}\,\varphi \leftrightarrow \triangle^{\mathrm{T}}\,\varphi$$

$\Diamond$

From the rules of temporal necessitation for the irreflexive operators above we can derive the corresponding rules for the reflexive operators:

PROPOSITION A.34 whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \boxminus^{\mathrm{T}}_{=\tau}\,\varphi$ $\Diamond$

PROPOSITION A.35 whenever $\vdash_{\mathrm{LTL}} \varphi$ then $\vdash_{\mathrm{LTL}} \boxplus^{\mathrm{T}}_{=\tau}\,\varphi$ $\Diamond$

DISTRIBUTED STATIC PROPERTIES

Recall from chapter 4 that LTL-formulae denoting *distributed static* properties only contain locative temporal connectives with a superscript L such that the temporal reference point will be fixed during evaluation of these formulae.

As for the metric temporal operators above (see propositions A.7 and A.8) we also get equivalences for the metric locative operators:

PROPOSITION A.36

$$\boxdot^{\mathrm{L}}_{=\eta}\,(\varphi_1 \wedge \varphi_2) \leftrightarrow (\boxdot^{\mathrm{L}}_{=\eta}\,\varphi_1 \wedge \boxdot^{\mathrm{L}}_{=\eta}\,\varphi_2)$$

$\Diamond$

PROPOSITION A.37

$$\Diamond^{L}_{=\eta}\,(\varphi_1 \wedge \varphi_2) \leftrightarrow (\Diamond^{L}_{=\eta}\,\varphi_1 \wedge \Diamond^{L}_{=\eta}\,\varphi_2)$$

◊

PROPOSITION A.38

$$\nabla^{L}\boxdot^{L}\varphi \leftrightarrow \boxdot^{L}\varphi$$

◊

PROPOSITION A.39

$$\neg\Diamond^{L}\varphi \leftrightarrow \boxdot^{L}\neg\varphi$$

◊

DISTRIBUTED DYNAMIC PROPERTIES

Recall from chapter 4 that LTL-formulae denoting *distributed dynamic* properties contain locative temporal connectives with a superscript L as well as T.

PROPOSITION A.40

$$\Diamond^{T}\boxdot^{L}\varphi \rightarrow \boxdot^{L}\Diamond^{T}\varphi$$

◊

PROPOSITION A.41

$$\Diamond^{T}\Diamond^{L}\varphi \leftrightarrow \Diamond^{L}\Diamond^{T}\varphi$$

◊

PROPOSITION A.42

$$\Diamond^{T}\Diamond^{L}\varphi \leftrightarrow \Diamond^{L}\Diamond^{T}\varphi$$

◊

We conclude this appendix with proposition A.43 containing a more general version of a property that has been formulated and used in section 6.1.2 to characterize message diffusion in distributed real-time systems. Because the proof of theorem 6.1 has been given there independent of the properties of the corresponding predicate the proof of the proposition here will be analogous.

PROPOSITION A.43 Let $\varrho \in \mathbb{ID}_T$ be some arbitrary but fixed temporal distance.

$$\forall \zeta : 0 < \zeta \quad \rightarrow \quad \left[\; \boxplus^{LT} [\, p \rightarrow \boxdot^{L}_{=1} \diamondsuit^{T}_{\leq \varrho} \, p \,] \right. \\ \rightarrow \\ \left. [\, p \rightarrow \boxdot^{L}_{=\zeta} \diamondsuit^{T}_{\leq (\zeta \otimes \varrho)} \, p \,] \right]$$

$\diamondsuit$

Bibliography

[1] W.J.H. Andrewes. *Time and Clocks.* In: The Astronomy and Astrophysics Encyclopedia, St.P. Maran (ed.), Van Nostrand Reinhold, 1992.

[2] R.C. Backhouse. *Program Construction and Verification.* Prentice-Hall International (UK) Ltd., London, 1986.

[3] J.C.M. Baeten and J.A.Bergstra. *Real Space Process Algebra.* In: LNCS 527, J.C.M. Baeten and J.F. Groote (eds.), Proceedings CONCUR '91, Springer-Verlag, 1991.

[4] J.C.M. Baeten and J.A.Bergstra. *Asynchronous Communication in Real Space Process Algebra.* In: LNCS 571, J. Vytopil (ed.), Proceedings of the 2nd International Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", Springer-Verlag, 1991.

[5] H. Barringer, R. Kuiper, A. Pnueli. *Now you may compose temporal logic specifications.* In: Proceedings of the 16th ACM Symposium on "Theory of Computing", pp. 51–63, 1984.

[6] H. Barringer, M. Fischer, D. Gabbay, G. Gough, R. Owens. *MetateM: A Framework for Programming in Temporal Logic.* Technical Report Series, UMCS-89-10-4, Department of Computer Science, University of Manchester, England, 1989.

[7] J. van Benthem. *Correspondence Theory.* In: Handbook of Philosophical Logic, D. Gabbay and F. Guenthner (eds.), Vol. II, pp. 167–247, D. Reidel Publishing Company, 1984.

[8] J. van Benthem. *Time, Logic and Computation.* In: LNCS 354, Proceedings of an International Conference on "Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency", J.W. de Bakker, W.P. de Roever, G. Rozenberg (eds.), Springer-Verlag, 1989.

[9] J. van Benthem. *The Logic of Time.* Second edition, Reidel, Dordrecht, 1991.

[10] M. Ben Ari, Z. Manna, A. Pnueli. *The Temporal Logic of Branching Time.* In: Acta Informatica 20, pp. 207–226, 1983.

[11] E.W. Beth. *Formal Methods.* D. Reidel Publishing Company, Dordrecht, Holland, 1962.

[12] R.M. Burstall. *Program Proving as Hand Simulation with a Little Induction.* In: Information Processing 74, Rosenfeld (ed.), Proceedings of

the IFIP Congress 74 in Stockholm, North-Holland Publishing Company, 1974.

[13] K.M. Chandy and J. Misra. *Parallel Program Design: A Foundation.* Addison-Wesley Publishing Company, Inc., 1988.

[14] C.C. Chang and H.J. Keisler. *Model Theory.* In: Studies in Logic and the Foundations of Mathematics, Vol. 73, North-Holland Publishing Company, 1973.

[15] B.F. Chellas. *Modal Logioc: An Introduction.* Cambridge University Press, 1980.

[16] P. Coesmans and M.J. Wieczorek. *Formal Specification of Fault Tolerant Real Time Systems Using Minimal 3-Sorted Modal Logic.* In: LNCS 571, J. Vytopil (ed.), Proceedings of the 2nd International Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", Springer-Verlag, 1991.

[17] F. Cristian. *A Rigorous Approach to Fault-tolerant Programming.* In: IEEE Transactions on Software Engineering, Vol. SE-11, No. 1, January 1985.

[18] F. Cristian, H. Aghili, R. Strong. *Clock Synchronization in the Presence of Omission and Performance Faults, and Processor Joins.* In: Proceedings of the 16th International Conference on "Fault-Tolerant Computing", IEEE, 1986.

[19] F. Cristian. *Agreeing on Who is Present and Who is Absent in a Synchronous Distributed System.* In: Proceedings of the 18th International Conference on "Fault-Tolerant Computing", IEEE, 1988.

[20] F. Cristian, H. Aghili, R. Strong, D. Dolev. *Atomic Broadcast: From Simple Message Diffusion To Byzantine Agreement.* Research Report RJ 5244 (rev.), IBM Research Division, 1989.

[21] F. Cristian, D. Dolev, R. Strong, H. Aghili. *Atomic Broadcast in a Real-Time Environment.* In: LNCS 448, Proceedings of a Workshop on "Fault-Tolerant Distributed Computing", B. Simons, A. Spector (eds.), Springer-Verlag, 1990.

[22] F. Cristian. *Reaching Agreement on Processor Group Membership in Synchronous Distributed Systems.* Research Report RJ 5964 (rev.), IBM Research Division, 1990.

[23] P.J. Davis and R, Hersh. The Mathematical Experience. Birkhäuser Boston, 1981.

[24] E.W. Dijkstra. *Hierarchical Ordering of Sequential Processes.* In: Acta Informatica 1, pp. 115–138, Springer-Verlag, 1971.

[25] E.W. Dijkstra. *A Discipline of Programming.* Prentice-Hall, Inc., 1976.

[26] L. Fariñas. *Space as Time.* In: Information Processing Letters 17, pp. 113–115, Elsevier Science Publishers b.v. (North-Holland), 1983.

[27] R.W. Floyd. *Assigning Meaning to Programs.* In: Proceedings of Symposia in "Applied Mathematics", AMS, Vol. 19, pp.19–32, 1967.

[28] D. Gabbay. *Investigations in Modal and Tense Logics with Applications to Problems in Philosophy and Linguistics.* Reidel, Dordrecht, 1976.

[29] D. Gabbay. *Two-Dimensional Propositional Tense Logic.* In: "Language in Focus", A. Kasher (ed.), pp. 569–583, Reidel 1976.

[30] D. Gabbay. *Labelled Deductive Systems, Part I.* Centrum für Informations- und Sprachverarbeitung, Universität München, CIS-Bericht-90-22, Februar 1991.

[31] J.W. Garson. *A New Interpretation of Modality.* In: Journal of Symbolic Logic 34, p. 535, 1969.

[32] J.W. Garson. *Indefinite Topological Logic.* In: Journal of Philosophical Logic 2, pp. 102–118, D. Reidel Publishing Company, Dordrecht-Holland, 1973.

[33] J.W. Garson. *Quantification in Modal Logic.* In: Handbook of Philosophical Logic, D. Gabbay, F. Guenthner (eds.), Vol. II, pp. 249–307, D. Reidel Publishing Company, 1984.

[34] R. Goldblatt. *Diodorian Modality in Minkoski Spacetime.* In: Studia Logica 2–3, 1980.

[35] A. Goswami, M. Bell, M. Joseph. *ISL: An Interval Logic for the Specification of Real-Time Programs.* In: LNCS 571, Proceedings of the 2nd International Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", J. Vytopil (ed.), Springer-Verlag, 1991.

[36] F. Harary, R.Z. Norman, D. Cartwright. *Structural Models: An Introduction to the Theory of Directed Graphs.* Wiley 1965.

[37] F. Harary. *Graph Theory.* Addison-Wesley Series in Mathematics, Addison-Wesley Publishing Company, Inc., 1969.

[38] D. Harel. *First-Order Dynamic Logic.* LNCS 68, Springer-Verlag, 1979.

[39] D. Harel and A. Pnueli. *On the Development of Reactive Systems.* In: "Logics and Models of Concurrent Systems", K.R. Apt (ed.), pp. 477–498, Springer-Verlag, 1985.

[40] E. Harel, O. Lichtenstein, A. Pnueli. *Explicit Clock Temporal Logic.* In: Proceedings of the 5th IEEE Symposium on "Logic in Computer Science", pp. 402–413, 1990.

[41] C.A.R. Hoare. *An Axiomatic Basis for Computer Programming.* In: Communications of the ACM, Vol. 12, No. 10, pp. 576–580 and p. 583, 1969.

[42] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice-Hall International, UK, Ltd., 1985.

[43] J. Hooman. *Specification and Compositional Verification of Real-Time Systems.* LNCS 558, Springer-Verlag, 1991.

[44] G.E. Hughes and M.J. Cresswell. *A Companion to Modal Logic.* Methuen and Co. Ltd., 1984.

[45] N. Husberg. *High Level Distributed Transition Systems.* In: Proceedings of the International BCS-FACS Workshop on "Semantics for Concurrency", M.Z. Kwiatkowski, M.W. Shields, R.M. Thomas (eds.), Springer-Verlag, 1990.

[46] ISO/IEC. *Open Systems Interconnection.* International Standard, Information Technology—Vocabulary—, Part 26, ISO/IEC 2382-26:1993 (E/F), ISO/IEC Copyright Office, Switzerland, 1993.

[47] D.M. Jackson. *Specifying timed communicating sequential processes using temporal logic.* Technical Report PRG-5-90, Oxford University Programming Research Group, 1990.

[48] G.J. Klir. *An Approach to General Systems Theory.* New York, 1969.

[49] J.A.W. Kamp. *Tense Logic and the Theory of Linear Orders.* Ph.D. Thesis, University of California, Los Angeles, 1968.

[50] W.P. Koole. *RDSL—Requirements and Design Specification Language from a Logical Point of View.* Hand-out of a Lecture during a Workshop on "Fault-Tolerance: Paradigms, Models, Logics, Construction", de Plasmolen, The Netherlands, December 13-14, 1990.

[51] H. Kopetz. *Accuracy of time measurement in distributed real-time systems.* In: Proceedings of the 5th Symposium on "Reliability in Distributed Software Database Systems", pp. 35–41, 1986.

[52] H. Kopetz and W. Ochsenreiter. *Clock Synchronization in Distributed Real-Time Systems.* In: IEEE Transactions on Computers, Vol. C-36, No. 8, 1987.

[53] R. Koymans, J. Vytopil, W.-P. de Roever. *Real-Time Programming and Asynchronous Message Passing.* In: Proceedings of the 2nd ACM Symposium on "Principles of Distributed Computing", August 1983.

[54] R. Koymans. *Specifying Message Passing and Time-Critical Systems with Temporal Logic.* LNCS 651, Springer-Verlag, 1992.

[55] L. Lamport. *Time, Clocks, and the Ordering of Events in a Distributed System.* In: Communications of the ACM, Vol. 21, No. 7, July 1978.

[56] L. Lamport. *Specifying Concurrent Program Modules.* In: ACM Transactions on Programming Languages and Systems, Vol. 5, No. 2, pp. 190–222, April 1983.

[57] L. Lamport and P.M. Melliar-Smith. *Synchronizing Clocks in the Presence of Faults.* In: Journal of ACM, Vol. 32, No. 1, pp. 52–78, January 1985.

[58] G. Le Lann. *Distributed Systems—Towards a Formal Approach.* In: Information Processing 77, B. Gilchrist (ed.), IFIP, North-Holland Publishing Company, 1977.

[59] R. de Lemos, A. Saeed, T. Anderson. *Analysis of Timeliness Requirements in Safety-Critical Systems.* In: LNCS 571, J. Vytopil (ed.), Proceedings of the 2nd International Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", Springer-Verlag, 1991.

[60] S.-T. Levi and A.K. Agrawala. *Real-Time System Design.* McGraw-Hill Publishing Company, 1990.

[61] O. Lichtenstein, A. Pnueli, L. Zuck. *The glory of the past.* In: LNCS 193, Proceedings of the International Conference on "Logics of Programs", pp. 196–218, Springer-Verlag, 1985.

[62] Z. Manna and A. Pnueli. *Axiomatic Approach to Total Correctness of Programs.* Memo AIM-210, Department of Computer Science, Stanford University, 1973.

[63] Z. Manna and A. Pnueli. *The Anchored Version of the Temporal Framework.* In: LNCS 354, Proceedings of an International Conference on "Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency", J.W. de Bakker, W.P. de Roever, G. Rozenberg (eds.), Springer-Verlag, 1989.

[64] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems—Specifications—.* Springer-Verlag, 1992.

[65] J.C. Maxwell. *Matter and Motion.* Dover, New-York, 1953.

[66] M. McConnell. *Challenger 'A Major Malfunction'.* Guild Publishing, London, 1987.

[67] J.M.E. McTaggart. *The Nature of Existence.* Cambridge University Press, 1927.

[68] W.H. Newton-Smith. *The Structure of Time.* Routledge and Kegan Paul, London, 1980.

[69] A. Pnueli. *The Temporal Logic of Programs.* In: Proceedings of the 18th Symposium on "Foundations of Computer Science", pp. 46–57, 1977.

[70] A. Pnueli. *The Temporal Semantics of Concurrent Programs.* In: Theoretical Computer Science 13, pp. 1–20, 1981.

[71] A. Pnueli and E. Harel. *Applications of Temporal Logic to the Specification of Real Time Systems.* In: LNCS 331, Proceedings of a Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", M. Joseph (ed.), pp. 84-98, Springer-Verlag, 1988.

[72] B. Randell, P.A. Lee, P.C. Treleaven. *Reliability Issues in Computing System Design.* In: ACM Computing Surveys, Vol. 10, No. 2, June 1978.

[73] W. Reisig. *Das Verhalten verteilter Systeme.* In: Gesellschaft für Mathematik und Datenverarbeitung, Bericht Nr. 170, 1987.

[74] N. Rescher and J. Garson. *Topological Logic.* In: The Journal of Symbolic Logic, Vol. 33, No. 4, December 1968.

[75] N. Rescher and A. Urquhart. *Temporal Logic.* Springer-Verlag, 1971.

[76] M. de Rijke. *The Modal Logic of Inequality.* In: ITLI Prepublication Series, LP-90-15, Institute for Language, Logic and Information, University of Amsterdam, 1990.

[77] W.-P. de Roever. *The Quest for Compositionality: A Survey of Assertion-Based Proof Systems for Concurrent Programs, Part I: Concurrency based on Shared Variables.* In: Proceedings of the IFIP Working Conference 1985: "The Role of Abstract Models in Computer Science", pp. 181–207, North-Holland, 1985.

[78] H.L. Roydan. *Real Analysis.* 2nd Edition, The Macmillan Company, 1968.

[79] H. Schepers. *Terminology and Paradigms for Fault-Tolerance.* In: "Formal Techniques in Real-Time and Fault-Tolerant Systems", J. Vytopil (ed.), pp. 3–31, Kluwer Academic Publishers, 1993.

[80] K. Segerberg. *Two-Dimensional Modal Logic.* In: Journal of Philosophical Logic 2, pp. 77–96, D. Reidel Publishing Company, Dordrecht-Holland, 1973.

[81] B. Simons, J. Lundelius Welsh, N. Lynch. *An Overview of Clock Synchronization.* In: LNCS 448, Proceedings of the Asilomar Workshop on "Fault-Toelrant Distributed Computing", B. Simons, A. Spector (eds.), Springer-Verlag, 1990.

[82] M. Sloman and J. Kramer. *Distributed Systems and Computer Networks.* Prentice-Hall 1987.

[83] R. Stuhlmann-Laeisz. *Many-Dimensional Topological Modal logic.* In: Logique et Analyse, September 1987.

[84] R. Swinburne. *Space and Time.* Macmillan, London, 1968.

[85] W.M. Turski. *Time Considered Irrelevant for Real-Time Systems.* In: BIT 28, pp. 473–486, 1988.

[86] W.T. Tutte. *Graph Theory.* Encyclopedia of Mathematics and its Applications, Vol. 21, Addison-Wesley Publishing Company, Inc., 1984.

[87] Y. Venema. *Many-Dimensional Modal Logic.* Ph.D. Thesis, University of Amsterdam, 1992.

[88] M.J. Wieczorek and J. Vytopil. *Specification and Verification of Distributed Real-Time Systems.* In: Proceedings of the 2nd International Conference on "Reliability and Robustness of Engineering Software", C.A. Brebbia, A.J. Ferrante (eds.), Computational Mechanics Publications 1991.

[89] M.J. Wieczorek and W.P. Koole. *Two-Sorted Modal Logic and its Application to Distributed Real-Time Systems.* In: Pre-Proceedings of the Applied Logic Conference "Logic at Work", CCSOM, University of Amsterdam, The Netherlands, 1992.

[90] M.J. Wieczorek. *Structured Specification of Distributed Real-Time Systems using Topological Locative Temporal Logic.* In: Proceedings of the 1st International Conference on "Software Quality Management", M. Ross, C.A. Brebbia, G. Staples, J. Stapleton (eds.), Computational Mechanics Publications and Elsevier Science Publishers, 1993.

[91] M.J. Wieczorek and W. van der Hoek. *Locative Temporal Logic: Soundness and Completeness.* (in preparation)

[92] A. van Wijngaarden et al. *Revised Report on the Algorithmic Language ALGOL 68.* In: Acta Informatica, Vol. 5, 1975.

[93] N. Wirth. *Toward a Discipline of Real-Time Programming.* In: Communications of the ACM, Vol. 20, No. 8, August 1977.

[94] L. Wittgenstein. *Tractatus Logico—Philosophicus.* Translated by D.F. Pears and B.F. McGuinness, Routledge and Keegan Paul, 1961.

[95] P. Zhou and J. Hooman. *Formal Specification and Compositional Verification of an Atomic Broadcast Protocol.* In: Computing Science Notes 94/05, M. Rem, K.M. van Hee (eds.), Department of Mathematics and Computing Science, Eindhoven University of Technology, 1994.

[96] J. Zwiers. *Compositionality, Concurrency and Partial Correctness.* LNCS 321, Springer-Verlag, 1989.

INDEX

INDEX TO SYMBOLS

GENERAL INDEX

A

B

C

Samenvatting

Dit proefschrift gaat over een nieuwe formele methode, namelijk locatieve temporele logica (LTL), voor de specificatie en verificatie van gedistribueerde real-time systemen. Locatieve temporele logica is een twee-soortige modale logica met als soorten ruimte en tijd. Wat de tijd betreft, is LTL gebaseerd op de reeds bekende temporele logica. Wat de ruimte betreft is LTL gebaseerd op een zogenaamde locatieve logica om ruimtelijke eigenschappen te kunnen beschrijven. Het onderligende model is een twee-soortige Kripke-model waarbij het universum is gedefinieerd als het direct product van een tijdsdomein en een ruimtedomein. Verder zijn op het semantisch niveau zowel binaire relaties voor locatieve en temporele bereikbaarheid als locatieve en temporele afstandsfuncties met erbij behorende domeinen beschikbaar.

Op het syntactisch niveau bevat LTL naast de bekende propositie logische operatoren, ook specifieke locatieve temporele operatoren voor het redeneren over ruimte of tijd. De taal zelf is een eerste-orde predicaatlogische taal waarbij quantificatie over tijd en ruimte *niet* is toegestaan. Quantificatie over het corresponderende afstandsdomein is wel toegestaan. De locatieve temporele operatoren zijn metrisch in die zin dat een index bij een operator de afstand in de tijd of ruimte aanwijst. De metrieken zijn ingevoerd zodat het mogelijk is om preciese afstanden in tijd en ruimte te specificieren, omdat het toepassingsgebied van gedistribueerde real-time systemen dit vereist.

In dit proefschrift hebben we de locatieve temporele logica toegepast op een aantal paradigma's uit het gebied van gedistribueerde real-time systemen: *dining philosophers*, *distributed watchdog*, *point-to-point-based* en *diffusion-based communication*, *synchronous* en *asynchronous communication*, *atomic broadcast*, en *processor group membership*. De dinerende filosofen hebben we bestudeerd vanuit een theoretisch oogpunt: de eenvoud van dit voorbeeld, zoals gedefinieerd in dit proefschrift, maakt het mogelijk om de verschillenden talen, te weten eerste orde predicaatlogica, temporele logica, locatieve logica en locatieve temporele logica, te vergelijken. Op deze manier worden de voor- en nadelen van de desbetreffende taal duidelijk, i.v.m. de specificatie van eigenschappen uit het toepassingsgebied. Alle andere paradigma's hebben we bestudeerd vanuit een practisch gezichtspunt: hoe geschikt is locatieve temporele logica voor de specificatie van gedistribueerde real-time systemen als de problemen niet triviaal zijn?

Een structureel defect van op klassieke logica gebaseerde specificatieformalismen (zie hoofdstuk 1) is dat geen expliciet onderscheid gemaakt kan worden tussen locale en globale eigenschappen. Dit wordt in LTL opgelost door de locatieve soort. Door het paradigma van een *externe waarnemer in ruimte en tijd* (zie sectie 5.1) kunnen we dan wel onderscheiden tussen aan de ene kant locale en gedistribueerde eigenschappen en aan de andere kant statische en

dynamische eigenschappen.

In hoofdstuk 1 worden een aantal eigenschappen behandeld, waar een geschikte specificatiemethode tenminste aan moet voldoen: *formaliteit, abstractievermogen, expressiviteit, scheiding van belangen* en *eenheid in veelheid.* Abstractievermogen en expressiviteit definiëren *wat* geschiktheid is en scheiding van belangen en eenheid in veelheid definiëren *hoe* geschiktheid bereikt kan worden.

FORMALITEIT

De eigenschap van formaliteit voor specificatiemethoden is algemeen geaccepteerd en staat garant voor korte en krachtige specificaties en strikte bewijzen. Zonder formaliteit zijn de specificaties in het algemeen sterk afhankelijk van het begrip dat de auteur of waarnemer van het desbetreffende systeem heeft. De bijdrage van formaliteit van een specificatiemethode tijdens de ontwikkeling van een specifiek systeem wordt gegeven door minstens twee aspecten: ten eerste formele verificatie van systeemeigenschappen en ten tweede hulp bij een beter begrip van systeemeigenschappen (validatie).

De specificatiemethode zoals geïntroduceerd in dit proefschrift bestaat uit een formele taal met een formele semantiek en een erbij behorend deductiesysteem (zie hoofdstuk 4 en appendix A). Deductie in LTL is gebaseerd op een binaire relatie die het volgende uitdrukt: *de formule aan de rechterkant is afleidbaar t.o.v. ruimte en tijd van de formule (of verzameling van formules) aan de linkerkant van het relatiesymbol.*

ABSTRACTIEVERMOGEN

Abstractievermogen maakt het mogelijk om alle niet-relevante details buiten beschouwing te laten. Evenals formaliteit is deze eigenschap algemeen geaccepteerd voor specificatiemethoden.

Een gedistribueerd real-time systeem kan vrijblijvend als volgt worden gedefiniëerd (zie sectie 1.2):

> *Een gedistribueerd real-time systeem is een (berekenings) systeem dat bestaat uit een aantal verschillende processen die mogelijk verspreid zijn over een aantal ruimtelijk gescheiden zelfstandige processoren en die aan bepaalde tijdseisen moeten voldoen. Om een gemeenschappelijk doel te bereiken, coördineren de processen hun activiteiten en gaan ze zo nodig informatie uitwisselen.*

Gedistribueerde real-time systemen hebben dus gemeenschappelijk de locatieve eigenschappen—gegeven door "mogelijk verspreid zijn over een aantal

ruimtelijk gescheiden zelfstandige processoren"—en de temporele eigenschappen—gegeven door "aan bepaalde tijdseisen moeten voldoen"—.

D.m.v. de locatieve temporele operatoren van LTL hebben we de mogelijkheid om locatieve en temporele eigenschappen onafhankelijk van een concrete implementatie te specificeren. Dit wordt duidelijk gemaakt door het paradigma van dinerende filosofen (zie de specificaties in elk van de hoofdstukken van deel I): alle formules in locatieve temporele logica hebben geen referentie meer nodig naar het aantal filosofen of naar specifieke filosofen, zoals bijvoorbeeld in temporele logica waar de filosofen wel bij hun naam moeten worden genoemd.

EXPRESSIVITEIT

Expressiviteit garandeert dat alle relevante details gespecificeerd kunnen worden. Evenals formaliteit en abstractievermogen is deze eigenschap algemeen geaccepteerd voor specificatiemethoden.

Uit de informele definitie van een gedistribueerd real-time systeem (zie onder ABSTRACTIEVERMOGEN) weten we dat zowel locatieve als temporele eigenschappen van belang zijn. Dus moeten dit soort eigenschappen in ieder geval gespecificeerd kunnen worden. Ter illustratie hebben we m.b.v. LTL in deel II van dit proefschrift enkele—ook niet triviale—problemen bestudeerd.

Hoofdstuk 7 bevat een discussie over het probleem van atomic broadcast, formele specificaties van de service en een klasse van protocollen in LTL. Bovendien hebben we een bewijs geleveerd dat de desbetreffende klasse van protocollen voldoet aan de eigenschappen gegeven door de service specificatie.

In hoofdstuk 8 wordt het probleem van processor group membership bestudeerd. Ook in dit geval geven we formele specificaties in LTL van de service en een specifieke klasse van protocollen die dan ook weer voldoen aan de service specificatie.

SCHEIDING VAN BELANGEN

Scheiding van belangen is een structurele eigenschap van specificatieformalismen en betekent dat het desbetreffende belang in de specificatiemethode op voorhand expliciet zichtbaar wordt. Deze eigenschap is in sterke mate afhankelijk van het toepassingsgebied en kan worden bereikt op verschillende manieren.

In LTL staan locatieve temporele operatoren ter beschikking die het mogelijk maken om vier belangrijke klassen van eigenschappen syntactisch te karakteriseren (zie sectie 4.1.1): locale statische eigenschappen die noch in de ruimte noch in de tijd kunnen veranderen, locale dynamische eigenschappen

die niet in de ruimte maar wel in de tijd kunnen veranderen, gedistribueerde statische eigenschappen die wel in de ruimte maar niet in de tijd kunnen veranderen en gedistribueerde dynamische eigenschappen die zowel in de ruimte als in de tijd kunnen veranderen.

EENHEID IN VEELHEID

Net zoals scheiding van belangen, is eenheid in veelheid een structurele eigenschap van specificatieformalismen. Het betekent dat de desbetreffende eenheid in de specificatiemethode op voorhand expliciet zichtbaar wordt. Deze eigenschap is in die zin complementair aan scheiding van belangen dat een bepaalde relatie tussen de gescheiden en misschien volledig onafhankelijke belangen tot stand gebracht moet worden. Zoals scheiding van belangen is ook eenheid in veelheid in sterke mate afhankelijk van het toepassingsgebied en kan worden bereikt op verschillende manieren.

Unificatie in logisch gebaseerde specificatieformalismen word meestal bereikt door een geschikte definitie van de deductierelatie. In LTL is deze relatie zowel afhankelijk van de ruimte als van de tijd.

Curriculum Vitae

Martin J. Wieczorek

geboren	op 22 april 1958 te Bochum (Duitsland)
getrouwd	sinds 13 september 1984
vrouw	Claudia U.E. Wieczorek, met meisjesnaam Lehmann, geboren op 15 juli 1959 te Bochum
kinderen	Christina C. Wieczorek, geboren op 4 februari 1986 te Starnberg Sarah A. Wieczorek, geboren op 30 maart 1988 te Bochum
woonplaats	D-44797 Bochum, Galgenfeldstraße 46

OPLEIDING

pasen 1965–1968	Basisschool (Katholische Volksschule Bochum)
1968–1977	Gymnasium (Goethe-Schule Bochum)
april 1977	Eindexamen gymnasium
oktober 77–augustus 84	Universiteit (Ruhr-Universität Bochum, RUB)
augustus 1984	Doctoraaldiploma (Dipl.-Math.) Wiskunde met bijvak Economische Wetenschappen, RUB

DIENSTVERBANDEN

oktober 1980–augustus 1984	student assistent rekencentrum RUB
september 1984–januari 1985	wetenschappelijk medewerker rekencentrum RUB
februari 1985–augustus 1986	wetenschappelijk medewerker Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (DFVLR, thans DLR), Oberpfaffenhofen
september 1986–mei 1991	wetenschappelijk medewerker rekencentrum RUB (vrijstelling van de plichten juli 1989–mei 1991)
juli 1989–januari 1992	toegevoegd onderzoeker bij de afdeling informatica aan de Katholieke Universiteit Nijmegen (KUN)
februari 1992–oktober 1993	projectmedewerker in dienst van de Nederlandse organisatie voor Wetenschappelijk Onderzoek (NWO)
november 1993–heden	medewerker in de groep Real-Time Systemen bij de afdeling informatica aan de Katholieke Universiteit Nijmegen (KUN)

Stellingen

behorende bij het proefschrift

Locative Temporal Logic
and
Distributed Real-Time Systems

—Specification—

van

Martin Josef Wieczorek

Nijmegen, 13 oktober 1994

1. Het paradigma van een *external observer in time*, die het gedrag van een computing systeem waarneemt en die vervolgens de gebeurtenissen die tijdens de executie optreden op een globaal tijdsschema projecteert, is wel geschikt voor sequentiële systemen, maar het is minder geschikt voor gedistribueerde real-time systemen (zie ook [W. REISIG]). Voor dit soort van systemen is het paradigma van een *external observer in space and time* geschikter.

   Zie hoofdstukken 1 en 5 van dit proefschrift.

2. Door meer *structuur* van de logische formules dragen specificaties in locatieve temporele logica (LTL) in het algemeen bij aan een beter begrip van eigenschappen van gedistribueerde real-time systemen. Een simpel voorbeeld kan gegeven worden door de eigenschap van eet-garantie van een groep van dinerende filosofen: terwijl de corresponderende eerste orde formule

   $$\forall p : \forall t : \exists t' : [t < t' \wedge eaten(p, t')]$$

   afhankelijk is van een variabele p over de groep van filosofen en van variabelen t en t' over een juist tijdsdomein, hebben we in LTL alleen een predicaat nodig, dat zonder parameters als volgt gedefinieerd kan worden:

   $$\boxdot^{\mathrm{L}} \boxplus^{\mathrm{T}} \diamondplus^{\mathrm{T}} eaten$$

   Zie hoofdstuk 4 van dit proefschrift.

3. De eigenschap van temporele *unieke identificatie* i.v.m. het verzenden en ontvangen van boodschappen [R. KOYMANS] kan d.m.v. LTL worden uitgebreid tot puur locatieve, puur temporele en locatieve temporele unieke identificatie. Dit uitgebreide concept kan dan worden gebruikt voor de specificatie van unieke identificatie van atomic broadcasts [F. CRISTIAN, H. AGHILI, R. STRONG, D. DOLEV].

   Zie hoofdstuk 7 van dit proefschrift.

4. *Kanalen* van een gedistribueerd real-time systeem zijn in LTL niet verbonden aan de locatieve bereikbaarheidsrelatie maar wel aan de corresponderende locaties. Op deze manier kunnen we van concrete kanalen abstraheren zonder de bereikbaarheid van locaties te verliezen. Zo nodig, kunnen we dan een domein van kanalen invoeren, waar we in de specificaties uitspraken over kunnen doen.

   Zie hoofdstukken 1, 4 en 6 van dit proefschrift.

5. Voor de formele specificatie van *fout-tolerante systemen* volstaat het gereedschap, dat ook voor niet fout-tolerante systemen gebruikt wordt. Een fout-hypothese is dan slechts een bijkomende aanname over eigenschappen van het desbetreffende (deel-)systeem.

   Zie hoofdstukken 7 en 8 van dit proefschrift.

6. De temporele chop-operator C, zoals toegepast b.v. in [H. BARRINGER, R. KUIPER, A. PNUELI] voor de specificatie van sequentiële compositie, kan als volgt worden uitgebreid:

$$\varphi_1 \; C_{\chi_1;\chi_2} \; \varphi_2$$

   De betekenis van deze formule is dat φ_1 geldig is totdat χ_1 waar wordt en vanaf dit tijdstip φ_2 geldig is totdat χ_2 waar wordt. De uitgebreide C-operator is nuttig voor de specificatie van fout-tolerante systemen waarbij een onderscheid gemaakt moet worden tussen normaal gedrag, zoals gegeven door φ_1, en exceptioneel gedrag, zoals gegeven door φ_2. De oorspronkelijke versie van de chop-operator kan dan als volgt worden gedefinieerd:

$$\varphi_1 \; C_{\varphi_2;\mathsf{T}} \; \varphi_2$$

   Zie pagina's 577–580 van [P. COESMANS, M.J. WIECZOREK].

7. De toepassing van *formele methoden* tijdens de ontwikkeling van computing systemen, met name gedistribueerde real-time systemen, dwingen af dat het desbetreffende systeem nauwkeurig in kaart wordt gebracht. Dit is niet alleen noodzakelijk voor de verificatie maar oók voor de validatie. Problemen die tijdens het formele bewijzen van eigenschappen op kunnen treden (verificatie) zijn soms te herleiden tot een discrepantie tussen de realiteit en de formele specificatie (validatie).

   Zie hoofdstuk 9 van dit proefschrift.

8. Zoals in [F.A. BROCKHAUS] vermeld staat kan men de *wetenschap* o.a. beschouwen als een "Prozeß methodisch betriebener, prinzipiell intersubjektiv nachvollziehbarer Forschung und Erkenntnisarbeit (Theorie und Praxis) auf Grund eines ursprünglichen, sachbestimmten Wissensdranges und Wahrheitssuchens."

   Tegenwoordig lijkt het meer op een soort "academische touwtrekkerij met verplichtingen in onderwijs en onderzoek".

9. Door een blind en onkritisch geloof in nieuwe *technologieën* kan men eenvoudige en natuurlijke probleemoplossingen, die even effektief zijn, over het hoofd zien. Met andere woorden: oplossingen voor complexe problemen hoeven niet zelf complex te zijn.

10. Als men zo spoedig mogelijk en met succes zijn studie wil beëindigen dan neemt de *hulpvaardigheid* voor andere mensen in grote mate af. Een gevolg daarvan is dat het principe van solidariteit in onze samenleving meer en meer verloren gaat. Dat is jammer!

11. *Chaos* is de normale en *orde* de buitengewone toestand. Meestal wordt een bepaalde orde kunstmatig tot stand gebracht door de mens en niet door de natuur.

REFERENTIELIJST

H. BARRINGER, R. KUIPER, A. PNUELI. *Now you may compose temporal logic specifications.* Proceedings of the 16th ACM Symposium on "Theory of Computing", pp. 51–63, 1984.

F.A. BROCKHAUS. *Brockhaus Enzyklopädie.* 17. völlig neubearbeitete Auflage, Band 20, F.A. Brockhaus, Wiesbaden, 1974.

P. COESMANS, M.J. WIECZOREK. *Formal Specification of Fault Tolerant Real Time Systems Using Minimal 3-Sorted Modal Logic.* Proceedings of the 2nd International Symposium on "Formal Techniques in Real-Time and Fault-Tolerant Systems", LNCS 571, Springer-Verlag, 1991.

F. CRISTIAN, H. AGHILI, R. STRONG, D. DOLEV. *Atomic Broadcast: From Simple Message Diffusion To Byzantine Agreement.* Research Report RJ 5244 (rev.), IBM Research Division, 1989.

R. KOYMANS. *Specifying Message Passing and Time-Critical Systems with Temporal Logic.* LNCS 651, Springer-Verlag, 1992.

W. REISIG. *Das Verhalten verteilter Systeme.* Bericht Nr. 170, Gesellschaft für Mathematik und Datenverarbeitung (GMD), Bonn, 1987.